أكسيس ٢٠٠٠
مع تطبيقات
ڤيچوال بيسك

Access 2000 VBA Handbook

Build Custom Databases and Projects with VBA

The Essential Resource for Power Users and Developers

On the CD:
- Sample databases
- VBA reference information







تأليف
سوزان نوفاليس

د. [illegible]ري

أكسيس ٢٠٠٠ مع تطبيقات فيجوال بيسك

Access 2000 VBA Handbook

دار الفاروق للنشر والتوزيع

أكبر مركز فى الشرق الأوسط
لإصدار أحدث الكتب فى عالم الكمبيوتر

العنوان

فرع وسط البلد: ٣ شارع منصور - المبتديان - متفرع من شارع مجلس الشعب محطة مترو سعد زغلول - القاهرة - مصر.

تليفون : ٧٩٥٣٠٣٢ (٠٠٢٠٢) - ٧٩٤٣٢٠٣ (٠٠٢٠٢)

فاكس : ٧٩٤٣٦٤٣ (٠٠٢٠٢)

فرع الدقى: ١٢ شارع الدقى الدور السابع - إتجاه الجامعة مَنْزَل كوبرى الدقى

تليفون : ٣٣٨١٠٢٢ (٠٠٢٠٢)

فاكس : ٣٣٨٢٠٧٤ (٠٠٢٠٢)

الطبعة العربية الأولى ٢٠٠٠

عدد الصفحات: ٨٨٠ صفحة

رقم الإيداع: ٣١٠٠ لسنة ٢٠٠٠

الترقيم الدولى: 977-307-086- 7

تاريخ الإصدار للنسخة الإنجليزية الأولى ١٩٩٩

أكسيس ٢٠٠٠ مع تطبيقات فيچوال بيسك

Access 2000 VBA Handbook

تأليف

سوزان نوفاليس

إعداد وترجمة

دار الفاروق و جى . إن . إس

لمحة على المحتويات

المحتويات

٤ الاتصال بالنماذج ١٨٥

٥ مبادئ برمجة VBA ٢٣٧

١٤ إنشاء وتعديل كائنات قاعدة البيانات ٧٤٩

١٥ توسيع أكسس ٨٠٥

المقدمة

- -

يُعد مايكروسوفت أكسس نظام إدارة قاعدة بيانات علاقية قيادية لإنشاء تطبيقات قاعدة بيانات على الشاشة الرئيسية. لماذا يُعد أكسس رقم واحد؟ هناك سببان وهما أن أكسس سهل دراسته وممتع في استخدامه. لقد حقق لمايكروسوفت نجاحاً كبيراً في توفير بيئة واجهة رسومية التي تجعلها سهلة لك لكي تتعلم استخدام القوة الكبيرة جداً المتوفرة في أكسس.

إذا فكرنا في استخدام أكسس تفاعلياً مثل السير، إذاً يتم تعلم كيفية وضع هذه البرامج معاً في تطبيقات مخصصة بسرعة. يوجد العديد من الكتب الممتازة الخاصة بالمقدمة التي تساعدك في تعلم كل شئ عن التعامل مع أكسس. ويُعد كتاب Mastering Access 2000 للكتّاب Elizabeth Olson و Alan Simpason "Sybex 1999" كتاب يساعدك بشكل خاص. يوجد القليل من الكُتيبات السريعة، واحد من أفضلها هو Access 2000 Developer's Handbook للكتّاب Ken Getz و Paul Litwin و Mike Gilbert "Sybex 1999". يُعد الكتاب الذي تقرأه الآن هو كُتيب التشغيل الذي يغلق الفجوة.

ماذا عن VBA في أكسس ٢٠٠٠

يوّفر مايكروسوفت VBA في أكسس ٢٠٠٠ كأداة قوية للتطور من أجل جعل قاعدة بياناتك آلية. يقوم مايكروسوفت بدمج VBA في هذا المنتج ليجعل أكسس ٢٠٠٠ نظام لإدارة قاعدة بياناتك قوي ومتعدد الجوانب لمستخدمي ومبرمجي الكمبيوتر الحاليين.

يحتاج المبرمجين القوة الإضافية والقدرة على التعامل مع الأخطاء التي تقدمها VBA. تسمح VBA للمبرمجين بالتحكم في أحداث معالجة ومواجهة المستخدم لإنشاء حل لقاعدة البيانات يكون وظيفي، مؤثر، وسهل للمستخدم.

ستتعلم في هذا الكتاب أساسيات برمجة Access VBA. وستتعلم أيضاً كيفية إنشاء إجراءات لثلاث فئات أساسية لعملية قاعدة البيانات: التنقل خلال التطبيق، الحفاظ على البيانات، وتحديد مجموعات التسجيلات للأغراض المُحددة.

لماذا يجب أن تقرأ هذا الكتاب

يُعد هذا الكتاب كتاب متوسط المستوى عن مايكروسوفت أكسس. يجب أن يكون عندك علم بالأفكار الأساسية وتقنيات أكسس التفاعلية، بما في ذلك إنشاء قاعدة بيانات بسيطة ومكتملة بالجداول المتصلة بها، الاستعلامات، النماذج، التقارير وصفحات تشغيل البيانات. يتم اعتماد هذا الكتاب على تلك المعلومات، كما يوضح لك كيف تجعل العمليات الخاصة بقاعدة البيانات آلية

باستخدام برمجة Access VBA. فلا تحتاج أي تجربة سابقة خاصة بالبرمجة. على الرغم من أن كتاب أكسس هذا يُعد متوسط المستوى إلا أنه كتاب برمجة لمستوى البادئ.

كيف يتم تنظيم هذا الكتاب

يتكوّن هذا الكتاب من خمسة عشر فصلاً ومفسّر للكلمات. يقوم الفصل الأول بشرح كيفية استخدام معالجي ومستخدمي أكسس ليجعل قاعدة البيانات آلية كمقدمة للأفكار التي تحتويها التطبيقات الآلية المنشأة. يقوم الفصل الثاني والثالث بتغطية العناصر الأساسية لبرمجة VBA الخاصة بالكائنات والأحداث. ويوفر لك مقدمة النوع كائن أكسس. ولأن النماذج تلعب دوراً أساسياً في تطبيقات أكسس، يكشف الفصل الرابع العديد من النقط الأساسية المتقدمة في تصميم النموذج.

يوفّر الفصل الخامس والسادس تغطية عميقة أنواع الكائنات المستخدمة في برمجة VBA. يقوم الفصل الخامس بإعادة زيارة أنواع الكائن أكسس، ويركز على الميزات المتوفرة فقط في VBA. يصف الفصل السادس Data Access Object "DAO" وأنواع ActiveX Data Object "ADO" التي تستخدم بواسطة أجهزة قاعدة البيانات.

تُغطي الثلاث فصول القادمة عملية كتابة الإجراءات الخاصة ببرمجة VBA. أما الفصل السابع فيقدم أساسيات كتابة الإجراءات، ويصف الفصل الثامن كيفية استخدام المتغيرات ويشرح الفصل التاسع كيفية التحكم في تنفيذ عبارات البرنامج.

تتعامل الفصول المتبقية مع التقنيات لاستخدام برمجة VBA لجعل مهمات قاعدة البيانات آلية. ففي الفصل العاشر، ستتعلم كيفية التعامل مع الأخطاء في VBA. كما تُغطي الفصول الحادية عشر والثانية عشر والثالثة عشر المهمات الهامة لتنقل قاعدة البيانات، وصيانة البيانات، ومعالجة التسجيلات. ويصف الفصل الرابع عشر كيفية إنشاء وتعديل كائنات قاعدة البيانات التي تستخدم إجراءات VBA. وأخيراً، يقدم الفصل الخامس عشر بعض التقنيات المتقدمة للتوسع في وظيفة أكسس، بما في ذلك استخدام الرابطة الحركية للمكتبات وتقنيات ActiveX.

كما يوفّر مفسّر الكلمات الموجود في آخر الكتاب قائمة أبجدية للمصطلحات في هذا الكتاب وتعريفاتها.

تنظيم كتابك

يُعد Access 2000 VBA Handbook مرجع وبرامج تعليمية يدوية. ففي معظم الفصول، ستقوم بإنشاء إما قاعدة بيانات جديدة من لا شئ أو إنشاء نسخة من نموذج Northwind لقاعدة البيانات.

١- قم بإنشاء مجلّد جديد تحت اسم VBA Handbook تخزّن فيه مثال قواعد البيانات الخاصة بك.

٢- حدد مكان مجلّد Samples. ففي تثبيت افتراضي مايكروسوفت أوفيس ٢٠٠٠، يكون الاتجاه C:\Program Files\Microsoft Office\Office\Samples.

٣- اسحب نسخ لكل الملفات الموجودة في مُجلّد Samples لمجلّد VBA Handbook الخاص بك. تُعد العديد من الملفات التي تقوم بنسخها من مجلد Samples صورة والملفات الأخرى التي تستخدمها قاعدة بيانات Northwind. يجب أن تكون هذه الملفات المتصلة في نفس المجلّد مثل النسخ التي تعمل بها الخاصة بقاعدة بيانات Northwind. تُعد الملفات التي ستحتاجها في هذه الفصول الآن متوّفرة في المجلد التي تعمل به.

ميزات جديدة في أكسس ٢٠٠٠

يوّفر لك أكسس ٢٠٠٠ الميزات المتعددة التي لم تكن موجودة في أكسس ٩٧. تُعد الميزات الجديدة التي يتم تغطية العديد منها في هذا الكتاب متضمنة لميزات تجعل من السهل إيجاد واستخدام المعلومات، تسمح ويب للميزات بالمشاركة في المعلومات، وأدوات تحليلية لإدارة المعلومات وتحسينات إضافية للبرمجة.

ميزات المعلومات

لقد تم إضافة أو زخرفة الميزات الآتية في أكسس ٢٠٠٠ لتسهيل إيجاد واستخدام المعلومات:

تحويل قاعدة البيانات إلى إصدار سابق: يمكن أن تحفظ قاعدة البيانات في إصدار سابق لأكسس، لتجعل من السهل مشاركة ملفات قاعدة البيانات مع مستخدمي الإصدارات المختلفة.

إطار قاعدة البيانات الجديد: يلائم إطار Database التي تم مراجعته الكائنات الجديدة لأكسس ٢٠٠٠ ويقوى القدرة على استخدامها. لقد تم تصميم إطار Database لتتماشى مع واجهة المستخدم الجديدة في أكسس ٢٠٠٠، وأيضاً خلال مجموعة أوفيس ٢٠٠٠ بأكملها.

اسم الاختصار: تحلل هذه الميزة آلياً معظم الآثار الجانبية العامة التي تظهر عندما يتم إعادة تسمية كائن قاعدة بيانات.

التنسيق الشرطي: توّفر لك هذه الميزة الدعم للأرقام السلبية والإيجابية والقيم التي يمكن التعبير عنها بأقل من أو أكثر من، بين ذلك أو مساوية إلى.

السحب والإفلات إلى إكسل: توّفر لك هذه الأوراق عرض صورة داخل صورة التي تسمح لك بالتركيز على وتحرير بيانات متصلة في نفس الإطار.

نموذج تحسينات: تجعل واجهة متطورة من السهل تعديل الحقول، مثل تغيير ألوانهم وخطواتهم، مباشرة من داخل عرض النموذج.

معالج علاقات الطباعة: يمكن أن تطبع الآن رسم بياني مرئي لإطار Relationships، التي تجعل من السهل رؤية كيفية هيكلة قاعدة البيانات.

تجميع عناصر التحكم: تسمح لك هذه الميزة بتجميع عناصر التحكم كوحدة فردية، التي تُسهل تصميم النموذج.

تقرير Snapshots المتطور: يمكن إنشاء تقارير Snapshots of Access 2000 وتوزيعها على القرص، مثل المستندات المطبوعة، في صفحة ويب أو كبريد إلكتروني.

الضغط: سيُحدّد أكسس ٢٠٠٠ آلياً عندما تكون ذات كفاءة لكي تضغط قاعدة البيانات ولتضغط قاعدة البيانات عندما يتم إغلاق الملف إذا كان التصغير في مسافة القرص بارزة.

تحديث قاعدة بيانات Northwind: توّفر قاعدة بيانات Northwind التي تعمل مع أكسس ٢٠٠٠ كنموذج تطبيق العديد من الأمثلة للميزات الأخيرة لك لكي ترى أو تنسخ.

ميزات تمكين ويب

لقد تم إضافة أو زخرفة الميزات الآتية في أكسس ٢٠٠٠ لتسمح بدرجة كبيرة من مشاركة المعلومات:

صفحات تشغيل البيانات: تسمح لك تلك الكائنات الجديدة بمد تطبيقات البيانات إلى الشبكة الداخلية عن طريق إنشاء صفحات HTML للبيانات المنضمة بسرعة وسهولة.

صفحات تشغيل مجموعة البيانات: توّفر لك هذه الميزة القدرة على عرض وإدارة المعلومات المتصلة في تنسيق هيكلي تم طيّه عند إنشاء صفحات تشغيل البيانات.

مربع أدوات صفحات تشغيل البيانات: يسمح لك مربع الأدوات الجديد في بيئة تصميم صفحة تشغيل البيانات بسحب وإفلات عناصر التحكم حسب احتياجك.

قائمة الحقل: تسمح لك هذه الميزة بإضافة المعلومات لعرض صفحة تشغيل البيانات عن طريق سحب وإفلات أسماء الحقل من قائمة سهل الوصول إليها.

تناول الارتباط التشعبي المتطور: يجعل واجهة الارتباط التشعبي المتطور من السهل إنشاء، تحرير، اتباع وإزالة الارتباطات التشعبية في قواعد البيانات.

أدوات التحليل

لقد تم إضافة أو تحسين الميزات الآتية في أكسس ٢٠٠٠ لتوّفر قدرات تحليل أفضل:

إمكانية التشغيل المتداخل لقاعدة البيانات المتطورة: يدعم أكسس ٢٠٠٠ OLE DB الذي يسمح لك بخلط سهولة استخدام واجهة أكسس مع قابلية تغيير الأبعاد لنهاية خلفية مشروعات قواعد البيانات.

مشروعات مايكروسوفت أكسس: يمكن لواجهة أكسس الآن أن تُنشئ نوع ملف جديد الذي يتصل مباشرةً لتخزين البيانات المتكاملة مما يجعل هذا سهلاً لك لكي تقوم بإنشاء تطبيقات العميل/الخادم مستخدماً واجهة أكسس المعروفة.

معالجي أكسس المتطورين: لقد تم تحديث العديد من معالجي أكسس لدعم مهمات جديدة لمشروع أكسس، مثل إنشاء قاعدة بيانات جديدة، تقرير أو نموذج.

أدوات تصميم المشروع: تسمح لك أدوات التصميم الجديدة أن تُنشئ بسهولة كائنات جانب الخادم، بما في ذلك الجداول، العروض، الإجراءات المُخزنة، والرسوم البيانية لقاعدة البيانات.

أدوات إدارة خادم SQL: يمتد أكسس ٢٠٠٠ للعديد من أنشطة إدارة قاعدة البيانات، مثل عمل نسخة مماثلة، نسخة احتياطية، إعادة تخزين وأمان لمايكروسوفت SQL الخاص بالخادم ٧.

تحسينات برمجية

يتم إضافة أو زخرفة الميزات الآتية في أكسس ٢٠٠٠ لتوفير البرمجة المتطورة:

دعم Unicode: تسمح هذه الميزة للشركات متعددة الجنسيات بدعم التطبيقات في إصدارات بلغة مختلفة.

VBA متطورة: يوّفر الإصدار الأخير VBA 6 العديد من الميزات الموجودة في نظام تطوير فيجوال بيسك.

الاصطلاحات المستخدمة في هذا الكتاب

يستخدم هذا الكتاب الاصطلاحات الآتية:

- تضغط على تركيبة المفاتيح المشار إليها عن طريق ضم المفاتيح بعلامة + "الجمع". على سبيل المثال، تشير Shift+F2 أنك تضغط باستمرار على مفتاح Shift بينما تضغط على مفتاح دالة F2.

- يتم الإشارة إلى أوامر تتابع القائمة بالرمز ⇦. على سبيل المثال، تشير File ⇦ Close إلى أمر Close على قائمة File.
- تظهر الكلمات، الجُمل، والأسماء التي يجب أن تكتبها أو تُدخلها في نوع الخط الأسود العريض.
- يستخدم نوع مسافة مونو بأمثلة برمجة الرمز. تظهر الكلمات الأساسية لعبارات SQL بأحرف كبيرة "على سبيل المثال DISTINCTROWC".

ملاحظة تمدك الملحوظة بمعلومات إضافية عن الموضوع.

تلميح دائماً تشير كلمة تلميح إلى طريقة أكثر كفاءة لتحقيق المهمة.

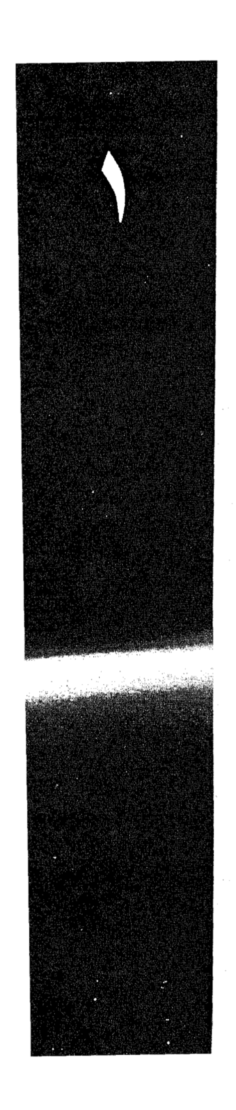
تحذير تنبهك التحذيرات لمشاكل يمكن أن تواجهها.

مربعات مربع الفقرات الإضافية

توّفر لك المقاطع الموضوعة في مربع تفسير لمواضيع مُعيّنة تتصل بالمناقشة المحيطة. على سبيل المثال، يتضمن الفصل ٥ في أساسيات VBA على مربع فقرات إضافية عن التغليف، ويتضمن الفصل ١٥ في أكسس الممتد على مربع فقرات إضافية عن الحصول على عناصر تحكم ActiveX.

تعليقات ختامية

شكراً على اختيارك لهذا الكتاب كي يساعدك على تعلم برمجة Access VBA. يُعدّ هذا الكتاب فرصة جيدة جداً لك لكي تتعلم المزيد عن البرمجة ولتشارك رؤيتي. أتمنى أن تستمتع بدراسة هذا الكتاب. أرجو أن تُرسل تعليقاتك، اقتراحاتك وتصحيحاتك إلى S@sfu.edu في Susann Novails أو إلى dana@mcwtech.com في Dana Jones.



جعل قاعدة البيانات آلية

بدون برمجة

تُعد قاعدة البيانات تجميع للسجلات والملفات. لكي تقوم بإنشاء قاعدة بيانات تحتاج نظام يساعدك على تخزين استرداد وتنويع بياناتك بالإضافة إلى تحليل وتحويل تلك البيانات إلى معلومات مفيدة. إذا كانت قاعدة البيانات كبيرة ومعقدة، من المحتمل أن تريد استخدام تطبيق قاعدة بيانات كمبيوتر تجاري مثل مايكروسوفت أكسس.

يمتلك أكسس مجموعة أدوات ومعالجين ممتازة لتساعدك على إنشاء قاعدة بيانات بما في ذلك "Tables" لتخزين البيانات "Quarries" لاسترداد ومعالجة البيانات، النماذج لإدخال وعرض البيانات "Data Access Pages" للعرض والعمل مع البيانات الموجودة على الإنترنت أو على الشبكة الداخلية وReports لطباعة المعلومات. ولكن إذا توقفت عند هذه النقطة ستستفيد فقط بكسر من القوة التي يمنحها أكسس، ستستخدم فقط خمسة من سبع كائنات محتوى قاعدة البيانات وتترك Modules وMacros دون أن تلمسهم.

تُعد قاعدة البيانات Interactive بدون ماكرو أو وحدات نمطية. ففي بيانات تفاعلية يبدأ المستخدم كل إجراء فردي يقوم به الكمبيوتر باختيار أمر قائمة أو بنقر زر شريط الأدوات. يقوم المستخدم بإجراء الاتصال بين النماذج والتقارير في قاعدة البيانات. لكي تقوم بالمهمات، يحتاج المستخدم أن يعرف أي من أوامر القائمة يستخدمها وأي تتابع يستخدمهم فيه بالإضافة إلى كيفية ترابط النماذج والتقارير. ففي قاعدة بيانات تفاعلية يكون للمستخدم تحكم كامل. يكون للمستخدم المُلم بجميع المعلومات القوة لكي يفسد البيانات ويتلف قاعدة البيانات بتحديد أمر الخطأ في الوقت الخطأ.

ستتعلم في هذا الكتاب كيف تُحوّل قاعدة البيانات التفاعلية الخاصة بك إلى Automated database Application. يمكن لأي مستخدم أن يستخدم تطبيق قاعدة بيانات آلية ومُصممة جيداً. لا يحتاج المستخدم أن يعرف تتابع خطوات المهمة أو أوامر أكسس. يحتاج المستخدم فقط أن ينقر زر فردي لينفذ مهمة صعبة.

عندما تقوم بإنشاء تطبيق جعلته آلياً تماماً، تُنشئ واجهة مخصصة للمستخدم. تُعد User Interface هي ما يراه المستخدمين على الشاشة وكيف يستخدمون لوحة المفاتيح والماوس ليتصل بالكمبيوتر. ففي واجهة المستخدم الخاصة بالتطبيق المُخصص، ينقر المستخدم أزرار الأمر ليتحرك بين المهمات. قم بعمل عمليات إدخال البيانات، إيجاد السجلات وطباعة التقارير. تُعد واجهة المستخدم المُخصصة هي المكان الذي يوجد فيه المستخدم بتطبيق قاعدة البيانات الخاصة به. فمن منظور المستخدم، تُعد واجهة المستخدم المخصصة هي تطبيق قاعدة البيانات الخاصة بك.

عندما تقوم بإنشاء واجهة جديدة، يجب أن تدعّم الأدوات لكي تفتح النماذج، تقوم بإدخال البيانات، تُحدد مكان سجلات مُعينة أو مجموعات من السجلات، بيانات الاستيراد، أرشيف

السجلات القديمة، والتقارير المطبوعة. يجب أن توفر أيضاً اختيار المسارات للتنقل خلال قاعدة بياناتك. وتتأكد من أن المستخدم يعرف دائماً مكان وجودهم وكيف تقوم بالتراجع إلى المسار.

يوفر لك أكسس مجموعة معالجين ومساعدين يساعدوك ببعض الآلية. يقدم لك هذا الفصل Database Wizard لإنشاء المسودة الأولى للتطبيق الكامل، Combo Box Wizard وCommand Button Wizard لإنشاء أزرار الأمر ومربعات التحرير والسرد الآلية وSwitchboard Manager لإنشاء خرائط الطرق للنماذج والتقارير الموجودة في التطبيق.

يوضح لك هذا الفصل أيضاً كيف تستخدم تقنية الارتباط التشعبي من تقنية الإنترنت للتنقل بين كائنات قاعدة البيانات. ستتعلم كيف تستخدم الارتباطات التشعبية لتنتقل مباشرةً من نموذج في تطبيقك إلى أي مستند في نظام ملف الكمبيوتر الخاص بك أو في أي كمبيوتر آخر متصل بالكمبيوتر الخاص بك مستخدماً شبكة عمل الإنترنت TCP/IP.

تُعد أهدافك عند كل مرحلة من بناء واجهة المستخدم هي تضمين سهولة الاستخدام، الفهم البديهي وحماية التطبيقات. يوّضح هذا الفصل كيفية إنشاء قوائم مُخصصة وأشرطة أدوات حتى يوّفر تطبيقك الأدوات والأوامر فقط التي يحتاجها المستخدم. وستتعلم كيف تحمي تطبيقك بكلمة مرور وكيف تعين شروط بدء التشغيل فيتم تهنئة المستخدم الذي اجتاز اختبار كلمة المرور بنموذج تطبيق بدء التشغيل والقوائم المُخصصة وأشرطة الأدوات الخاصة بها.

ينتهي هذا الفصل بمعاينة VBA "Visual Basic for Application"، لغة البرمجة القوية المستخدمة في أكسس ويعطيك فكرة سريعة عن القوة الإضافية التي ستحصل عليها عندما تتعلم استخدامها.

استخدام معالج قاعدة البيانات

يمكن أن يساعدك Database Wizard على إنشاء تطبيقات قاعدة بيانات لعدد من الأعمال المختلفة والسيناريوهات الشخصية. تُعد بعض منها كالآتي:

Asset Tracking	Ledger
Contact Management	Order Entry
Event Management	Resources Scheduling
Expenses	Service Call Management
Inventory Control	Time and Billing

بمجرد أن تقوم بتعريف السيناريو القريب من التطبيق الذي تريد إنشاؤه، يعرض المعالج في نمط المعالج المعتاد سلسلة من الشاشات التي تخبرك عن التطبيق وإدخالك. فعقب تجميع اختياراتك، يستخدم المعالج القالب الذي قمت بتحديده لإنشاء وتخصيص الجداول، الاستعلامات، النماذج، التقارير، صفحات تشغيل البيانات والوحدات النمطية اللازمة.

ملاحظة تُعد صفحات تشغيل البيانات ميزة جديدة في أكسس ٢٠٠٠ الذي تسمح للمستخدمين بمد تطبيق قاعدة البيانات إلى الشبكة الداخلية للشركة عن طريق إنشاء صفحات HTML للبيانات التي تم طيها بسهولة وسرعة. يساعد هذا المستخدمين في المشاركة السريعة والأكثر كفاءة للمعلومات.

يمكن لقاعدة بيانات المعالج أن تنشئ قواعد بيانات سهلة ومعقدة. يمكن للمعالج أن يقوم بإنشاء العديد من مجموعات الجداول معتمداً على السيناريو الذي تختاره عندما يكون هناك أزواج من الجداول في العديد والعديد من العلاقات. يقوم المعالج بالحل الآلي للعلاقة داخل زوج من واحد إلى العديد من العلاقات عن طريق إنشاء جدول علاقة. يقوم المعالج بإنشاء نماذج لكل جدول ويمكن حتى أن يُنشئ تركيبة من نموذج فرعي ليعرض علاقة واحدة أو أكثر. يُنشئ المعالج تقارير تلخيص مناسبة للسيناريو الذي اخترته.

إنشاء مسارات نقل عن طريق Switchboards

فعقب إنشاء تقارير التلخيص ونماذج إدخال البيانات الفردية، يقوم المعالج بإنشاء نماذج تسمى Switchboards، التي توّفر لك مسارات نقل بين مجموعات النماذج والتقارير. يُنشئ المعالج Main Switchboards ليخدم كمركز تحكم للتطبيق. لقد أمـر Main Switchboards الأزرار لكلاً من المهمات الأساسية لقواعد البيانات. يأخذك النقر على زر Main Switchboards إلى نموذج تستخدمه لإنجاز مهمة قاعدة البيانات، مثل إدخال البيانات داخل واحد من الجداول. يمكن للنقر على زر Main Switchboards أن يأخذك أيضاً إلى لوحة تبديل أخرى بأزرار تأخذك إلى نماذج وتقارير أخرى ولوحات تبديل أخرى. يوضح الشكل ١-١ على مسارات نقل لوحة التبديل.

تتفاعل الأزرار عندما تنقرهم لأن المعالج قد أنشأ مجموعة فردية من التعليمات لكل زر. يكتب المعالج التعليمات ويخزنهم في واحد من المكانين:

- ففي Standard Modules الموجودة بالقائمة ككائنات منفصلة في لوح Modules لإطار Database.

♦ ففي Report Modules و Form Modules التي تم إنشائه داخل النماذج والتقارير "كجزء من تعريف النموذج أو التقرير"، تخزّن كجزء من النموذج الموجود بالقائمة في حول Forms أو التقرير المُدرج بالقائمة في حول Reports الخاصة بإطار Database.

الشكل ١-١

التصال لوحة التبديل.

Main Switchboard
Data Entry
Summary Reports
Exit Application

Data Entry
Form

Reports Switchboard
First Report
Second Report
Go to Main Switchboard

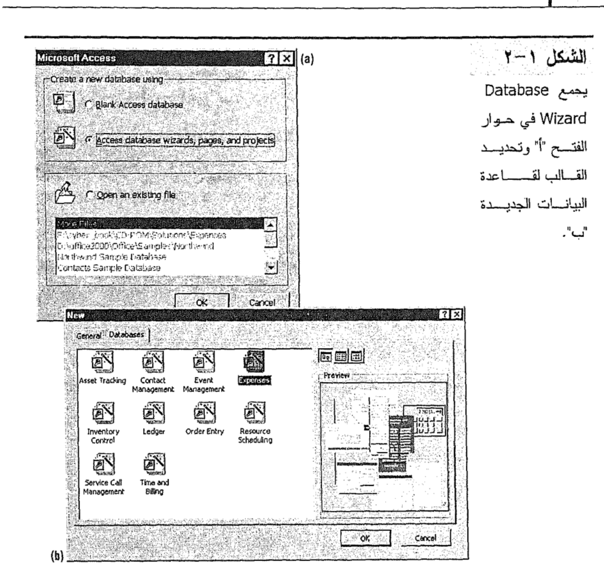
Data-Entry Form

Second Report

First Report

لكي تلاحظ Database Wizard أثناء عملها، ستقوم بإنشاء تطبيق لتعقب مصاريف الموظفين.

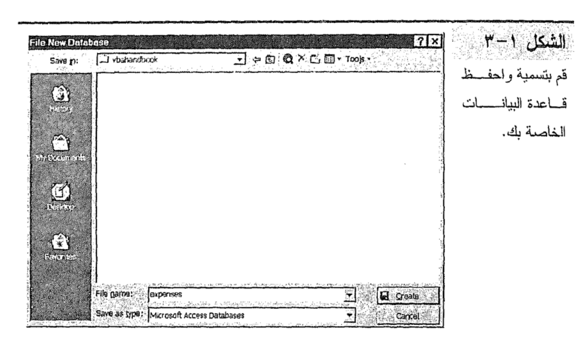
١- ابدأ بتشغيل أكسس ٢٠٠٠ وانقر أزرار رادير Access Database Wizard، Pages، و Projects "راجع الشكل ٢-١ أ". انقر OK وبعد ذلك انقر جدول Databases في مربع حوار New واختيار Expenses كقالب لاستخدامه في إنشاء قاعدة البيانات الجديدة الخاصة بك "راجع الشكل ٢-١ ب".



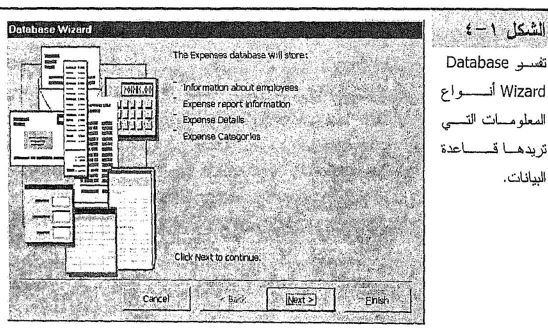
الشكل ١-٢

يجمع Database Wizard في حوار الفتح "أ" وتحديد القالب لقاعدة البيانات الجديدة "ب".

٢- ففي الحوار التالي، ادخل Expenses كاسم واحفظ قاعدة البيانات إلى مجلد VBA Handbook "راجع الشكل ١-٣". إذا لم تنشئ هذا المجلد، راجع المقدمة للتعليمات عن إعدادها. انقر زر Create لكي تبدأ Database Wizard. تفسر الشاشة الأولى للمعالج أنواع معلومات قاعدة البيانات التي ستديرها.



الشكل ١-٣
قم بتسمية واحفظ قاعدة البيانات الخاصة بك.

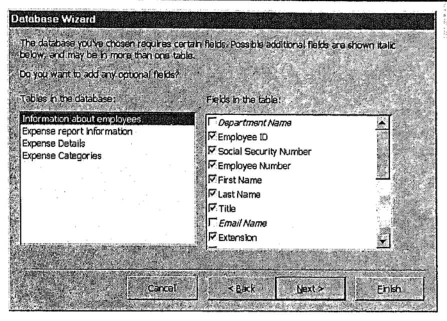


الشكل ١-٤
تفسر Database Wizard أنواع المعلومات التي تريدها قاعدة البيانات.

٣-تعطيك الشاشة القادمة على الفرصة التي تُمكنك من عمل التغييرات الدقيقة في قاعدة البيانات "راجع الشكل ٥-١". يعرض مربع القائمة في اليسار الجداول التي سيتم إنشائها. عندما ننقر على جدول، يتغير مربع القائمة في اليمين ليعرض الحقول الخاصة بالجدول المحدد. لا يمكن أن تضيف جداول جديدة أو تقوم بإلغاء جداول من القائمة، ولكن يمكن أن تضيف حقول واضحة في الخط المائل. دقق في الحقول التي تريد أن تضيفها في كل جدول.

الشكل ١-٥

يمكن أن تختار إضافة حقول اختيارية لبعض الجداول وتعمل على ازدحام قاعدة البيانات بنموذج البيانات.

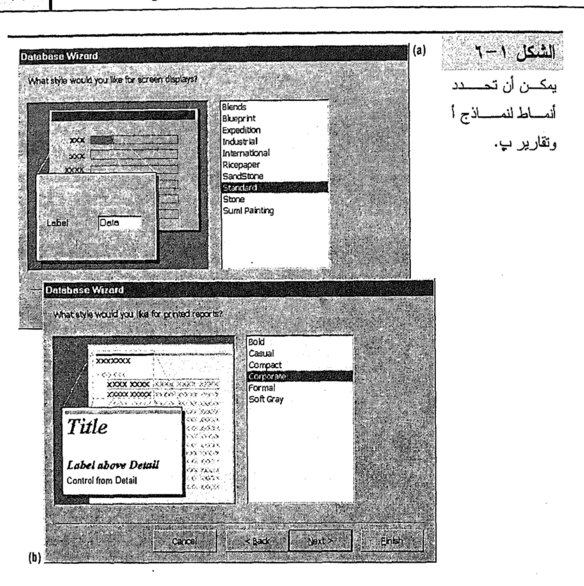


٤- حدد الأنماط للنماذج "راجع الشكل ٦-١ أ" والتقارير "راجع الشكل ٦-١ ب" في الشاشتين القادمتين.

٥- يمكن أن تستخدم الشاشة القادمة لإدخال عنوان لقاعدة البيانات وتقوم بتضمين صورة الصور النقطية "راجع الشكل ٧-١ أ". إذا أضفت صورة، ستظهر على التقارير التي يقوم المعالج بإنشائها. ففي الشاشة النهائية "راجع الشكل ٧-١ ب"، يمكن أن تحدد بدء قاعدة البيانات مباشرةً عقب إنشائها وتعرض المساعدة. يجعل النقر على Finish المعالج مستعداً للعمل. فبينما يُجتهد المعالج، يعرض الحوار متر واحد من التقدم يوضح التقدم الشامل وأمتار تقدم أخرى توضح التقدم في إنشاء كائن معين.

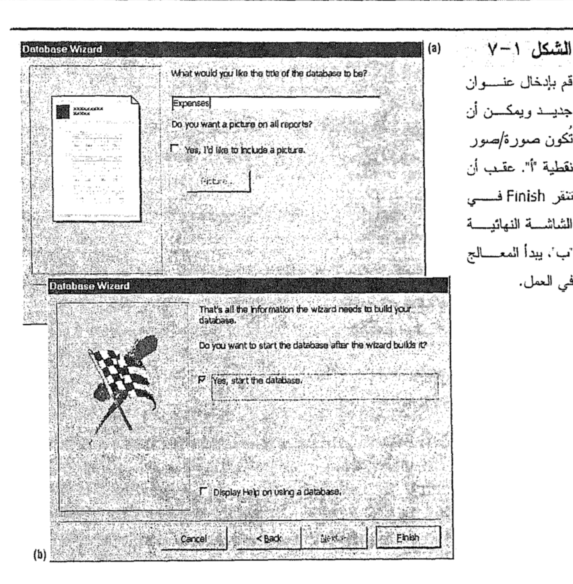
الشكل ١-٦

يمكـن أن تحـــدد أنمـاط لنمـــاذج أ وتقارير ب.



تحـذيـر

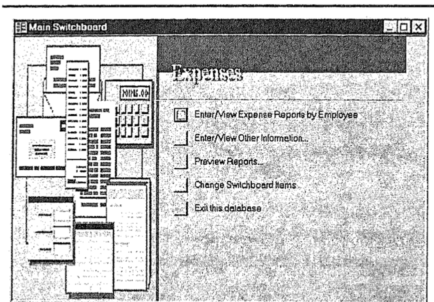
إذا لم تكن الطابعة على الكمبيوتر الخاص بك، سينتج أكسس خطأ يشير إلى أنها لن تنشئ جيداً قاعدة البيانات. ففي هذا المثال، حـــدد فقـط Start ⇦ Settings ⇦ Printers والنقر المزدوج للطابعة Add. سيعمل هذا علـى بدء تشغيل Wizard لطباعة Add.



الشكل ١-٧

قم بإدخال عنوان جديد ويمكن أن تكون صورة/صور نقطية "أ". عقب أن تنقر Finish في الشاشة النهائية "ب"، يبدأ المعالج في العمل.

يُنشئ المعالج الجداول والعلاقات، نموذج يسمى Switchboard والنماذج والتقارير. يوضح النص الأخير الذي يومض فوق متر التقدم الأسفل أن المعالج سيقوم بإعداد خصائص قاعدة البيانات. عندما ينتهي العمل، يتم عرض Main Switchboard "راجع الشكل ١-٨"، ويتم تصغير إطار Database.



الشكل ١-٨

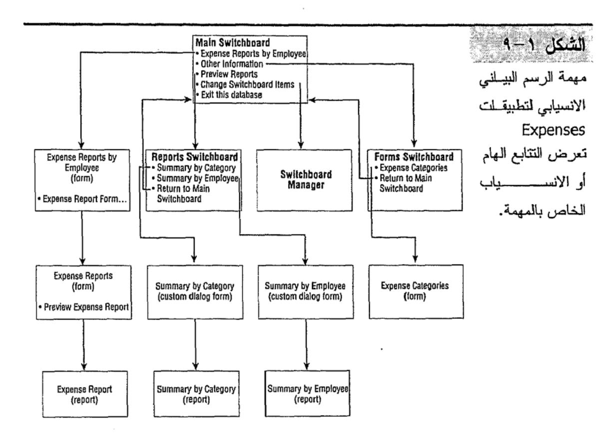
يُعد Main Switchboard الإنجاز الرئيسي لتطبيق قاعدة البيانات.

استكشاف التطبيق

تُعد Main Switchboard الوقفة الأولى في التنقل خلال التطبيق. "ستتعلم كيفية استخدام Main Switchboard كنموذج بدء تشغيل فيما بعد في هذا الفصل". يعطيك Main Switchboard معنى فوري للمهمات الأساسية التي يديرها التطبيق. توجهك أزرار الأمر إلى نماذج لإجراء مهمات قاعدة البيانات أو للوحات تبديل أخرى يمكن أن يتفرع لنماذج ولوحات تبديل أكثر من ذلك. يظهر الزرين الآخرين على كل Main Switchboard التي تقوم بإنشائها Database Wizard. ستكتشف زر Change Switchboard Items فيما بعد في هذا الفصل. إذا نقرت Exit سيغلق زر قاعدة البيانات بدون وجود أكسس.

خذ دقائق قليلة لتصل إلى نماذج أخرى عن طريق نقر الأزرار. توّفر أزرار النماذج ولوحات التبديل مسارات تنقل خلال التطبيق إلى المهمات المختلفة. يظهر التنظيم الشامل كما هو واضح في مهمة الرسم البياني الانسيابية في الشكل ٩-١.

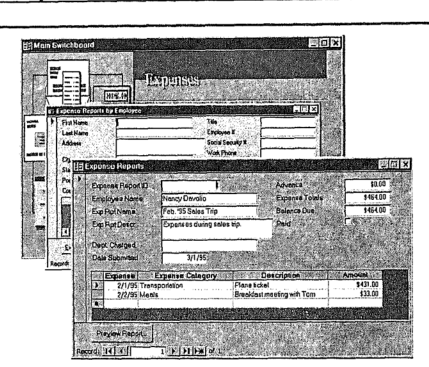
تشير اسم الاتجاهات في رسم بياني خاص بالمهمة الانسيابية إلى كون النموذج له زر أمر يأخذك للخلف إلى نموذج سابق أم لا.



الشكل ١-٩

مهمة الرسم البياني الانسيابي لتطبيقات Expenses تعرض التتابع الهام أو الانسياب الخاص بالمهمة.

يأخذك الزر الأول على Main Switchboard إلى Expense Reports بواسطة نموذج Employee، الذي له زر يفتح نموذج Preview Reports. يفتح النقر على Expense Reports الخاص بنموذج Expense Reports معاينة لتقرير Expense Reports. سيأخذك نقر تتابع الأزرار على مسار اتجاه واحد من Main Switchboard إلى تقرير Expense Reports "راجع الشكل ١٠-١". يعد المسار اتجاه واحد لأنه لا يوجد أزرار للأمر تأخذك للخلف إلى Main Switchboard. "بالطبع، يمكن أن تستخدم زر نموذج Close الافتراضي في أعلى الركن الأيمن أو تختار أمر Close من قائمة File لغلق النموذج والرجوع إلى النموذج السابق." ففيما بعد، سنقوم بالتنقل للخلف إلى النموذج السابق بسهولة عن طريق إضافة أزرار الأمر إلى النماذج.



الشكل ١٠-١
يُعد مسار الاتجاه الواحد من Main Switchboard إلى نموذج Expense Reports إلى تقرير Expense Report.

لاحظ أنه عندما تُحدد موظف في Expense Reports بواسطة نموذج Employee وانقر الزر على النموذج، يُفتح نموذج Expense Reports بالمعلومات لنفس الوظيفة، وهذا يعني أن النموذج المفتوح يُعد Synchronized بالنموذج الذي يفتحها. إذا قمت بالنقر إلى النموذج الأول، انتقل إلى موظف مختلف وبعد ذلك انقر للخلف إلى النموذج الثاني، سترى أن النموذجين يبقوا متزامنين. وبالمثل، عندما تنقر زر Preview Reports الموجود على زر Expense Reports، يُعد التقرير الذي قد تم فتحه متزامن مع النموذج الذي قد قام بفتحه.

يأخذك الزر الثاني على Main Switchboard إلى لوحة تبديل أخرى تسمى Forms Switchboard، حيث يكون لديك اختيار عرض نموذج إدخال بيانات Expense Categories أو الرجوع إلى Main Switchboard "راجع الشكل ١-١١".

Expense Report

Expense Report

Date Submitted	3/1/95
Employee ID	1
Employee Name	Nancy Davolio
Exp Rpt Name	Feb. '95 Sales Trip
Dept. Charged	
Description	Expenses during sales trip.

Date	Description	Expense Category	Amount
2/1/95	Plane ticket	Transportation	$431.00
2/2/95	Breakfast meeting with Tom	Meals	$33.00

Page: 1

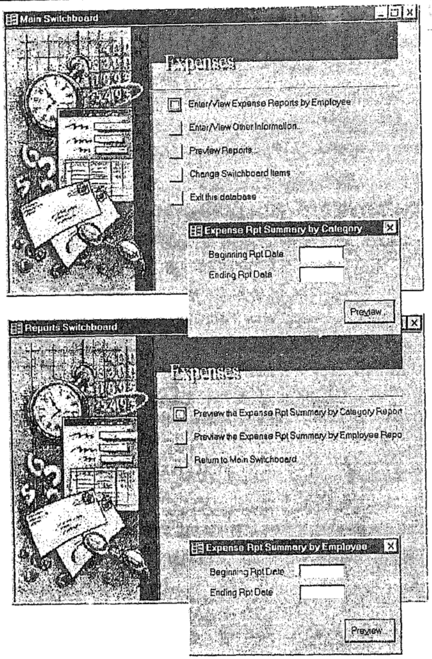
الشكل ١-١١
المسار من Main Switchboard إلى نموذج Expense Categories.

يتفرع الزر الثالث إلى لوحة تبديل ثالثة، تسمى Reports Switchboard حيث يمكن أن تنقر أياً من تقريرين التقارير أو الرجوع إلى Main Switchboard. يفتح النقر على أياً من أزرار تقرير التلخيص نموذج حوار مخصص حيث تُدخل تواريخ البداية والنهاية لتقرير ما (راجع الشكل ١-١٢". يأخذك نقر زر Preview عقب إدخال التواريخ إلى معاينة لتقرير التلخيص للفترة المُحددة.



الشكل ١-١٢

المسار من Main Switchboard إلى مربعات الحوار المُخصصة لتجميع إدخال التواريخ لتقارير التلخيص.



لقد قام Database Wizard بأكثر من توفير المسار التنقلي بين النماذج والتقارير. لقد أنشأ المعالج أيضاً نموذج حوار مُخصص لتجميع الإدخال. ففي هذا المثال، يُعد الفاصل الزمني للتاريخ هو المعيار لاستعلام المعامل الذي يُحدد السجلات المناسبة لتقرير التلخيص "استعلام المعامل الذي يحصل على المعلومات من نموذج يستخدم تقنية تسمى Query by Form، الذي ستتعرف المزيد عنها في الفصل ١٣".

اختيار عمل المعالج

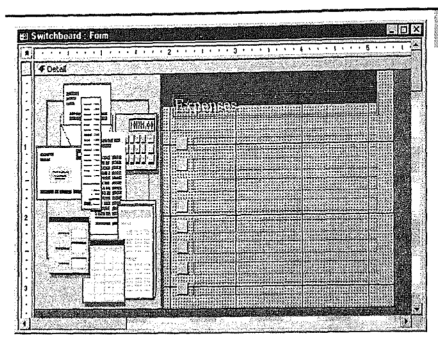
دعنا ننظر خلف المناظر لنستكشف كيف يحقق المعالج بعض من مهماتها. يستخدم Database Wizard عدد من التقنيات المتقدمة والعنصرية، ستتعلم استخدام بعضاً منها في تطبيقات قاعدة البيانات الخاصة بك.

أعد تخزين إطار Database "بواسطة تكبير إطار Expense" ولاحظ الآتي:

- هناك أيضاً نموذج واحد فقط يسمى Switchboard، Reports Switchboard، Forms Switchboard وMain Switchboard. إذا قمت بفتح Switchboard في عرض Design، سترى نموذج بثمان أزرار للأوامر وثمان عناوين فارغة "راجع الشكل ١٣". تستخدم Database Wizard هذا النموذج لجميع لوحات التبديل. يتم إنشاء كل لوحة تبديل بسرعة كإصدار مختلف لنفس النموذج عندما تنقر زر الأمر. يُحّول نقر الزر الثاني على Main Switchboard النموذج إلى Forms Switchboard ويُحّول نقر الزر الثالث النموذج إلى Reports Switchboard. لقد أنشأ المعالج تعليمات لتحويل النموذج بما في ذلك تغيير التعليق، عرض الرقم الصحيح للأزرار والعناوين وتمكين أزرار الأمر المعروض لكل إصدار للنموذج لينفذ مهماتهم المُحددة.

تلميح لكي تفتح كائن في عرض Design، حدد الكائن في لوح Objects لإطار Database عن طريق نقره، وبعد ذلك انقر زر Design.

- وبالإضافة إلى أربع جداول للبيانات، هناك جدول Switched Items. يحمل واحد من الحقول في هذا الجدول، حقل Item Text العناوين للأزرار على لوحات التبديل المتنوعة. تُخزّن الحقول الأخرى معلومات لإنشاء لوحات التبديل وجعل الأزرار تعمل. لاحظ أن هذا الجدول يُعد مصدر السجل لنموذج Switchboard "سيتم مناقشة مصادر السجل في الفصل ٦".



الشكل ١-١٣
عرض Design الخاص بنموذج Switchboard.

- على الرغم من أن المعالج قد استخدم تقنية Query By Form ليُحدد السجلات المستندة على إدخالك في نموذج الحوار، فلا يوجد أي استعلامات مُدرجة في Queries pane. ففي الواقع، لا يوجد هناك استعلامات مُخزنة مثل Saved Queries في تطبيق Expense. إذا كنت قد درست الطريقة التي أنشأ بها Report Wizard وForm Wizard كائناتهم، فستعرف أن تلك المعالجين قد استخدموا عبارة SQL بدلاً من استعلامات تم حفظها كمصادر للسجل خاصة بالنماذج والتقارير. تستطيع معظم التطبيقات التي يمكن Database Wizard أن ينشأ ما ليس لها أي استعلامات مُخزنة وتستخدم فقط عبارات SQL مباشرة للسجل ومصادر الصف.
- يوجد هناك نموذج Report Date Range فردي يستخدمه المعالج لكلاً من نماذج الحوار المُخصصة. لقد أنشأ المعالج تعليمات ليغير التعليق معتمداً على الزر الذي تنقره في Reports Switchboard.
- يوجد هناك وحدة نمطية قياسية فردية مسماة Global Code وهي مدرجة بالقائمة في لوح Models الخاصة بإطار Database. لا يوجد هناك Macros مدرج بالقائمة فيلوح Macros.

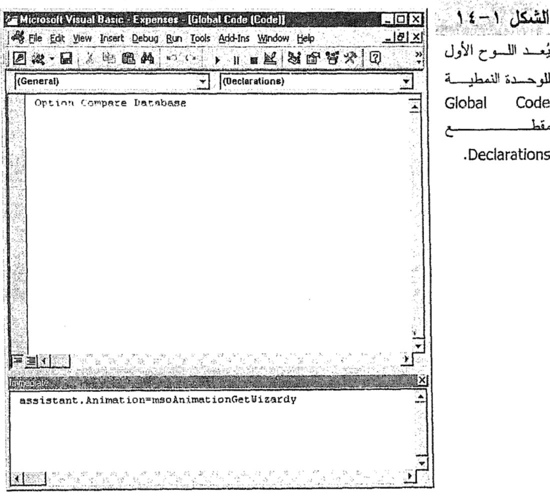
استكشاف الوحدة النمطية القياسية

دعنا نستكشف الوحدة النمطية القياسية. قم بنقرة مزدوجة للوحدة النمطية Global Code في لوح Module الخاص بإطار Database لتعرض إطار Module. يعتمد ما يتم عرضه في إطار على إعدادات اختيار إطار Module بالكمبيوتر الخاص بك. وغالباً سيتم فتح الإطار للوح الأولى

للوحدة النمطية "راجع الشكل ١-١٤" المسمى بمقطع Declarations. ففي هذا اللـــوح، تُخَــزن الاتجاهات الخاصة Access و"تعلن" أسماء الثوابت والمتغيرات وبعـــض الـــدوال التـــي تنـــوي استخدامها في الوحدة النمطية. يُخزّن مقطع Declarations عبارة لإعدادات الاختيار. إذا رأيـــت ص إضافي إذن فأنت تعرض عرض Full Module، انقر الزر في أقصـى أسفل اليسار لركـــن لإطار لتعرض عرض Procedure. استخدم ألواح وحدة نمطية متتابعة لتُخزّن مجموعات مـــن الإرشادات.

الشكل ١-١٤

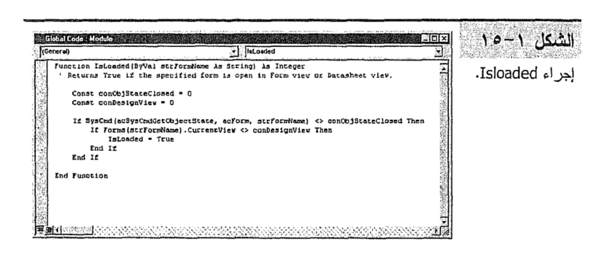
يُعـــد اللـــوح الأول للوحـــدة النمطيـــة Global Code مقطـــــــــع Declarations.



ملاحظـــة

يمكن أن تُغيّر العرض الافتراضي لإطار Module باختيار علامة تبويـــب Tools ⇦ Options ⇦ Editor ودقق أو امسح مربع حوار Default To Full Module. ولهذا الفصل، قم بمسح مربع الحوار لتعرض التعريفـــات والإجراءات في ألواح منفصلة.

تُخَزّن الوحدة النمطية الإرشادات المكتوبة في لغة برمجة VBA. ففي VBA، اكتب مجموعة من الإرشادات في وحدات تسمى Procedures. لكي ترى الإجراءات المُخَزّنة في الوحدات النمطية Global Code، انقر السهم الأسفل لمربع التحرير والسرد على اليمين. تشير قائمة مربع التحرير والسرد إلى أن الوحدة النمطية لها إجراء واحد فقط يسمى Isloaded "راجع الشكل ١-١٥".



الشكل ١-١٥
إجراء Isloaded.

على الرغم من أننا لم نتكلم بعد عن لغة VBA، فمن الواضح أن هذا الإجراء غير مسئول عن فتح النماذج والتقارير، بدون ذكر تزامنهم. كما ستتعلم فيما بعد في هذا الكتاب، يستخدم تطبيق Expense إجراء Isloaded ليحدد إذا كان هناك نموذج مُعَيّن مفتوح أم لا.

استكشاف الإجراء لزر الأمر



نحن نعرف أنه يجب أن يكون هناك إجراءات إضافية لأزرار الأمر، إذن سنستكشف الزر. افتح Expense Reports بواسطة نموذج Employee وقم بالتبديل إلى عرض Design. بعد ذلك حدد زر Expense Report Form وإذا كان ضرورياً افتح ورقة الخاصية الخاصة به عن طريق نقر زر Properties في شريط الأدوات.

تدرج ورقة خاصية الكائن في القائمة كل الخصائص التي تستطيع تعيينها وقت التصميم. بالإضافة إلى ذلك، يُعد لمعظم الكائنات خصائص لم يدرج بالقائمة في ورقة الخاصية. "ستعرف أشياء عن الخصائص التي لم تدرج بالقائمة في الفصول القادمة". تصف المجموعة الكاملة لخصائص الكائن في لحظة معينة وتُسمية State الخاصة بالكائن.

ملاحظة تُسَمّى وأنت في وضع التصميم وقت التصميم، في الوقت الذي يُعد فيه الإطار النشط عرض Design لو أحد من كائنات إطار Database.

عندما تُغيّر خاصية، تقوم بتغيير ولاية الكائن. على سبيل المثال عندما تنقر زر أمر ما، تُغيّر ولايته من كونه غير منقور. ففي أكسس، لقد تم إعطاء العديد من تغييرات الكائن في الولاية Events. وعندما تنقر زر أمر تتعرف على حدث Click. لكلاً من الأحداث التي تخصه. فالكائن له Event Property متطابقة أُدرجت بالقائمة في علامة تبويب Event الخاصة بورقة خاصية الكائن. يوضح الشكل ١-١٦ اثنى عشرة من خصائص زر الأمر.

الشكل ١-١٦
خصائص الحدث لزر الأمر.

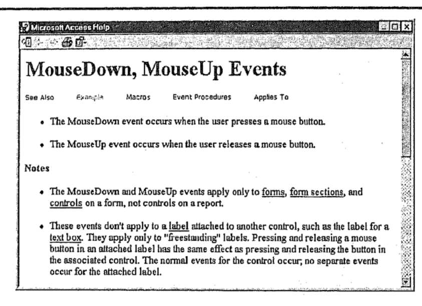
Command Button: ExpenseReport
Format | Data | Event | Other | All
On Enter
On Exit
On Got Focus
On Lost Focus
On Click [Event Procedure]
On Dbl Click
On Mouse Down
On Mouse Move
On Mouse Up
On Key Down
On Key Up
On Key Press

ففي معظم الحالات، يكون اسم خاصية الحدث هو كلمة "On" يتبعها اسم الحدث. على سبيل المثال، تتطابق خاصية حدث On Click مع حدث Click. يقترح اسم خاصية حدث ما الإجراء الذي يجعل ولاية الكائن تتغير. على سبيل المثال، يقترح On Mouse Down أنه عندما تضغط زر ماوس فبينما يكون المؤشر فوق الزر، يتعرف زر الأمر على حدث يسمى Mouse Down. وبينما يمكن أن تقوم باستنتاجات مماثلة عن خصائص حدث آخر وتكون صحيحاً معظم الوقت، فمن الأفضل أن تستثمر بعض الوقت باستخدام تعليمات فورية لتتعلم التعريف الدقيق لكل خاصية حدث. Click Help ⇦ What's This? ضع المؤشر فوق الخاصية، وانقر هذا يفتح Microsoft Access Help على الحدث الذي حددته "راجع الشكل ١-١٧". ستتعلم المزيد عن أحداث مُعيّنة وترتيب الأحداث في الفصل ٢.

تُعد الأسباب التي تجعل الأحداث مهمة هو أن الأحداث هي فرصة للبرمجة. يمكن أن تُنشئ قائمة للإرشادات وهذا يعني، A Program وتُخبر أكسس أن يُنفذ البرنامج عندما يتعرف الكائن على واحد من أحداثه. تسمى تقنية البرمجة هذه Event Driven Programming.

الشكل ١-١٧

التعليمات الفورية الخاصة بأحداث Mouse up وMouse Down.


MouseDown, MouseUp Events

See Also Example Macros Event Procedures Applies To

- The MouseDown event occurs when the user presses a mouse button.
- The MouseUp event occurs when the user releases a mouse button.

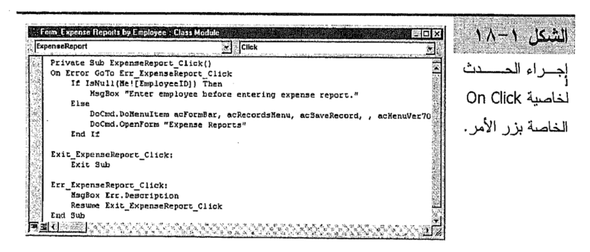
Notes

- The MouseDown and MouseUp events apply only to forms, form sections, and controls on a form, not controls on a report.
- These events don't apply to a label attached to another control, such as the label for a text box. They apply only to "freestanding" labels. Pressing and releasing a mouse button in an attached label has the same effect as pressing and releasing the button in the associated control. The normal events for the control occur; no separate events occur for the attached label.

يُوّضح الشكل ١-١٦ "Event Procedure" كإعداد لخاصية On Click. وهذا يعني أن إجراء VBA قد تم إنشاءه وإسناده إلى حدث زر Click. يقوم أكسس بتشغيل الإجراء عندما تنقر الزر. يُعد إجراء VBA الذي يتم تشغيله عندما يتعرف الكائن على حدث هو Event Procedure. فعندما تنقر زر Build على يمين خاصية حدث، يُفتح إطار Module ليظهر إجراء الحدث "راجع الشكل ١-١٨".

الشكل ١-١٨

إجراء الحدث لخاصية On Click الخاصة بزر الأمر.


Form_Expense Reports by Employee : Class Module

ExpenseReport | Click

```
Private Sub ExpenseReport_Click()
On Error GoTo Err_ExpenseReport_Click
    If IsNull(Me![EmployeeID]) Then
        MsgBox "Enter employee before entering expense report."
    Else
        DoCmd.DoMenuItem acFormBar, acRecordsMenu, acSaveRecord, , acMenuVer70
        DoCmd.OpenForm "Expense Reports"
    End If

Exit_ExpenseReport_Click:
    Exit Sub

Err_ExpenseReport_Click:
    MsgBox Err.Description
    Resume Exit_ExpenseReport_Click
End Sub
```

ستتعلم كيف تُنشئ إجراءات حدث VBA فيما بعد في هذا الكتاب. أما الآن، لاحظ أن شريط عنوان إطار Module يعرض تعليق Employee عن طريق Form_Expense مما يعني أننا نرى الوحدة النمطية للنموذج الخاصة Expense Reports عن طريق نموذج Employee. يُسّمى أكسس آلياً الوحدة النمطية للنموذج الذي يستخدم كلمة "Form" واسم النموذج المفصول بتسطير أسفل السطر، النقش العام، أو بناء الجملة يُعد Form_*formname*. وبالمثل، تُسمى أكسس آلياً الوحدة النمطية للتقرير باستخدام بناء جملة Report_*reportname*.

يُنشئ Database Wizard إجراءات حدث لكل زر من أزرار الأمر. تفتح بعض الإجراءات بسهولة النماذج أو التقارير الأخرى، والإجراءات الأخرى أيضاً تتزامن مع النموذج أو التقرير لتعرض سجل التطابق.

توفّر Database Wizard مسوّدة أولى مصممة جيداً للتطبيق. يمكن للشخص الذي يعرف أكسس أن يتنقل خلال مهماتها بسهولة. بينما لتطبيق Expenses الذي يتم وصفه *fully automated*. يجب أن يتطور بصورة أكبر حتى يستطيع الشخص الذي لا يعرف أكسس أن يستخدمه بدون إرشادات تفصيلية. دعنا نستكشف بعض مساعدي أكسس الآخرين الذي من الممكن استدعائهم لتصنيف ميزات تجعل التطبيق أسهل في استخدامهم.

استخدام Command Button Wizard

ينشئ Command Button Wizard آلياً أزرار الأمر وإجراءات الحدث VBA الخاصة بأحداث أزرار Click. يمكن لهذا المعالج القوى أن يُنشئ إجراءات للعديد من عمليات قاعدة البيانات المعروفة. يوّضح الجدول ١-١ ثلاثة وثلاثون إجراءاً يمكن أن تجعلهم آليين عن طريق Command Button Wizard.

الجدول ١-١ : Command Button Wizard بواسطة Operation Automated

الفئة	المهمة
Record Navigation	Find ،Previous Record ،Next ،Last ،Go to First .Find Next ،Record
Record Operation	،Duplicate Record ،Delete Record ،Add New Record .Undo Record ،Save ،Print Record
Form Operations	،Close Form ،Edit Form Filter ،Apply Form Filter Print Current ،Print a Form ،Open Page ،Open Form .Refresh Form Data ،Form
Report Operations	Send ،Mail Report ،Preview Report ،Print Report .Report to File

الجدول ١-١: Command Button Wizard بواسطة Operation Automated

الفئة	المهمة
Application	Run Notepad، Run MS Word، Run MS Excel، Quit Application، Run Application
Miscellaneous	Print Table، Run Macro، Run Query، Run AutoDialer.

يوّضح الرسم البياني الانسيابي للمهمة الواضح سابقاً في الشكل ٩-١ أن Database Wizard قد أنشأت مسار ذا اتجاه واحد من Main Switchboard إلى Expense Reports عن طريق نموذج Employee. لكي تجعل المسار الذي له اتجاهين بين النموذج Main Switchboard، ستنشئ زر أمر لتغلق Expense Reports عن طريق نموذج Employee وارجع إلى Main Switchboard.

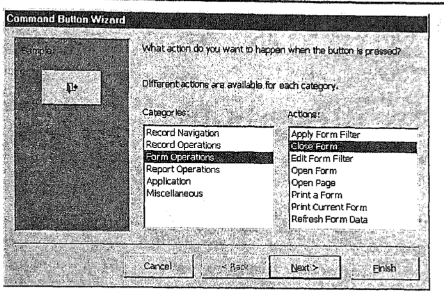


١- افتح Expense Reports عن طريق نموذج Employee في عرض Design. إذا لم يتم الضغط على زر مربع أدوات Control Wizard. انقر الزر الآن لتنشط Control Wizard.



٢- انقر أداة Command Button وبعد ذلك انقر في النموذج المجاور لزر Expense Reports توّضح الشاشة الأولى للمعالج قائمة بها ست فئات للمهمات وقائمة للإجراءات الخاصة بكل فئة.

٣- حدد فئة Form Operations وإجراء Close Form "راجع الشكل ١-١٩".
انقر زر Next.

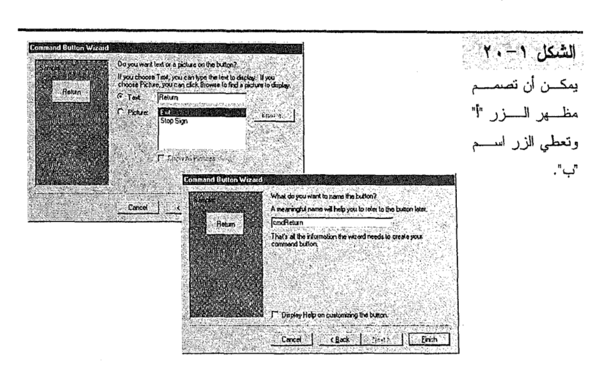


الشكل ١-١٩

يمكن Command Button Wizard أن يجعل عمليات قاعدة البيانات المعروفة آلية.

٤- ففي الشاشة التالية، تقوم بتصميم مظهر الزر. انظر نظرة ثاقبة وقم بإدخال Return كتعليق للنص بدلاً من استخدام صورة "راجع الشكل ١-٢٠ أ". انقر الزر Next.

٥- تعطيك الشاشة الأخيرة الفرصة لتسمي الزر. ادخل **cmdReturn** "راجع الشكل ١-٢٠ ب".



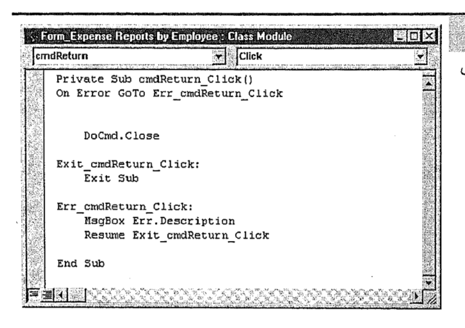
الشكل ١-٢٠

يمكن أن تصمم مظهر الزر "أ" وتعطي الزر اسم "ب".

٦- انقر زر Finish في الشاشة الأخيرة للمعالج. يقوم المعالج بإنشاء زر الأمر ويرفعه بإجراء VBA لينفّذ الإجراء الذي قمت بتحديده. دعنا نراجع الإجراء.

٧- تأكد من أنه تم فتح Expense Reports عن طريق نموذج Employee في عـرض Design. فعن طريق الزر المُحدد Return الذي أنشأ جديداً، انقـر زر Properties. انقر في خاصية حدث On Click وانقر زر Build على يمين مربع الخاصيـة. يظهر إجراء الحدث في إطارModule" راجع الشكل ١-٢١". يعرض مربع التحرير والسرد على اليسار اسم الزر، ويعرض مربع التحرير والسرد على اليمين اسم الحـدث. ففـي الفصول القادمة ستتعلم هدف كل سطر في الإجراء، أما الآن لاحـظ فقـط أن سطـر DoCmd.Close هو الإرشاد الذي يغلق النموذج.

٨- احفظ النموذج، حوّل للخلف إلى عرض Form، وانقر زر Return الجديد. يتم إغـلاق النموذج وترجع إلى Main Switchboard.



الشكل ١-٢١
إجراء الحدث لتغلق النموذج.

ملاحظـة

اختر أسماء الكائن بحذر. هناك سبب واحد لتحديد الاسم بحذر هو أنه غـير مقنع للغاية أن تُغيّر اسم الكائن فيما بعد. وهناك سبب أكثر أهمية وهـو أن هذه تُعد فرصتك لتقوم بعملك كمبرمج للتطبيق أسهل بكثير من أن تسـتخدم أسماء لها معنى. على سبيل المثال، باختيار الاسم cmdReturn بدلاً مـن قبول الافتراض "Command 7"، نحن نضع بالمستند أن الكائن يُعـد زر أمر باستخدام بادئة cmd "تُسمى tag في تسمية المرادف القياسي"، ونضـع بالمستند أن النقر على الزر يرجع إلى النموذج السابق عن طريق تضميـن كلمة Return كجزء موصوف "قم بتسمية *base name*". يناقش الفصل ٢ تسمية المقاييس.

استخدام الارتباطات التشعبية للتنقل

تُعد الارتباطات التشعبية ميزة تنقل جديدة في أكسس ٩٧، ولقد تم تحسينهم في أكسس ٢٠٠٠. تُعد *hyperlink* قطعة من النص، صورة، زر شريط أدوات، أمر قائمة أو زر أمر يمكن أن تنقره لتقفز إلى مكان آخر. ففي أكسس يمكن أن تستخدم ارتباطات تشعبية لتفتح وتقفز إلى الآتي.

- كائن آخر في قاعدة بياناتك.
- أي ملف متاح مخزون في نظام ملف الكمبيوتر الخاص بك "على الكمبيوتر الخاص بك أو على كمبيوتر آخر في شبكة عمل المساحة المحلية الخاصة بك".
- مكان فرعي محدد داخل أي Microsoft Office، بما في ذلك إشارة مرجعية في مستند وورد، نطاق في جدول بيانات إكسل، انزلاق في تقديم باور بوينت أو كائن في قاعدة بيانات أخرى خاصة أكسس.
- يمكن لأي ملف متاح في الشبكة الداخلية الخاصة لشركتك أو في الإنترنت العام. إذا كان الملف هو مستند HTML، أن تقفز إلى مكان فرعي مُحدد داخل المستند.
- نموذج بريد إلكتروني فارغ تم عنونته إلى عنوان بريد إلكتروني تحدده.
- لكي يعمل الارتباط التشعبي، يجب أن تستخدم التنسيق الصحيح لتعرف المكان الذي تريد أن تقفز إليه.

على سبيل المثال، لكي تعّرف ملف على الشبكة الداخلية أو على الإنترنت، استخدم عنوان Internet، يُسّمى (URL) *Uniform Resource Locator*. على سبيل المثال، http://www.microsoft.com يُعرّف الصفحة الرئيسية لموقع Microsoft Web على ftp://ftp.microsoft.com الذي يُعرّف موقع Microsoft ftp. ولكي تُعرّف ملف في نظام ملف الكمبيوتر الخاص بك، استخدم تنسيق قياسي لتحدد المسار للملف الذي يُسمى مسار (UNC) *Universal naming Convention*. على سبيل المثال، C:\Program Files\Microsoft Office\Access\Samples\Cajun.com يُعرّف مستند html الذي يمكن أن تثّبته كجزء من أكسس وC:\Program Files\Microsoft Office\Access\Samples\Northwind.mdb الذي يُعرّف قاعدة بيانات نموذج Northwind الخاص بأكسس. يُسمى الإنترنت أو عنوان مسار الملف *hyperlink adress*. يمكن أن تُحدد المكان داخل ملف Microsoft Office أو مستند HTML، الذي يسمى *hyperlink subaddress*. مستخدماً بناء الجملة الواضح في الجدول ٢-١.

الجدول ١-٢: بناء جملة العنوان الفرعي للارتباط التشعبي

نوع الملف	بناء الجملة الخاص بموقع داخل ملف
Microsoft Access	اسم كائن إطار Database. إذا كان هناك العديد من الكائنات بنفس الاسم، يضع أكسس الكائنات في الترتيب الآتي: نماذج وتقارير واستعلامات وجداول وصفحات بيانات أكسس وMacros ووحدات نمطية وعروض وأنظمة وإجراءات مُخزّنة وجداول SQL المرتبطة وtriggers. يمكن أيضاً أن تدخل *object type* *object name*. على سبيل المثال، لكي تحدد تقرير يسمى Suppliers، استخدم بناء الجملة Report Suppliers.
Microsoft Word	اسم الإشارة المرجعية. يجب أن تُعرّف الإشارة المرجعية في وورد قبل أن تقفز إليها.
Microsoft Excel	اسم النطاق. استخدم نطاق بناء جملة !Sheet. على سبيل المثال، لكي تُحدد الهدف كخلية L8 في ورقة العمل المسماة Source، استخدم بناء جملة Source!L8.
Microsoft Power Point	رقم الشريحة. على سبيل المثال، لتحديد الشريحة العاشرة، استخدم بناء الجملة ١٠.
مستند HTML	علامة Name.

عندما يكون هدف الارتباط التشعبي كائن آخر في قاعدة بيانات أكسس الحالية، فلا تحدد عنوان الارتباط التشعبي، فتحدد فقط العنوان الفرعي للارتباط التشعبي. على سبيل المثال، إذا كنت تعمل في قاعدة بيانات Expense وتريد أن تُنشئ ارتباط تشعبي مع نموذج Expense Categories كهدف، ستحتاج فقط أن تحدد العنوان الفرعي أو مثل Form Expense Categories.

إذا كان هدف الارتباط التشعبي كائن آخر في قاعدة بيانات أكسس التي تعمل معها حالياً، فسيفتح نقر الارتباط التشعبي في أكسس الكائن. يُفتح الجدول أو الاستعلام في عرض Datasheet ويتم فتح Macros والوحدات النمطية في عرض Design. إذا كان الهدف هو

ملف في تطبيق Microsoft Office آخر، يتم تصغير إطار أكسس الحالي ويتم فتح المثال الثاني الخاص بأكسس وبعرض الكائن الهدف. وإذا كان الهدف هو ملف في تطبيق Microsoft Office آخر، يتم تصغير إطار أكسس الحالي، ويتم فتح التطبيق ويعرض الهدف. أما إذا كان الهدف هو ملف أو مستند على الشبكة الداخلية أو الإنترنت، يتم تصغير إطار أكسس ويتم فتح المستعرض ليعرض المستند "يجب أن يتم فتح اتصال TCP/IP الخاص بك عندما تنقر الارتباط التشعبي".

يوّفر لك أكسس ٢٠٠٠ ثلاث طرق لتستخدم الارتباطات التشعبية للتنقل:

- يمكن أن تُخزّن عنوان الارتباط التشعبي في الجداول وتعرض الارتباطات التشعبية في ورقة بيانات أو في طيّ عنصر تحكم النموذج لحقل الارتباط التشعبي.
- يمكن أن تُنشئ ارتباط تشعبي في عنصر تحكم لم يتم طيّه على نموذج يستخدم عنوان، زر أمر أو صورة.
- يمكن أن تُحوّل أمر القائمة إلى ارتباط تشعبي.

تخزين الارتباطات التشعبية كبيانات في الجدول

يوّفر لك أكسس ٢٠٠٠ نوع بيانات Hyperlink لتخزين عناوين الارتباط التشعبي. على سبيل المثال، في تطبيق قاعدة البيانات ذات الإدخال المُرتب، يمكن أن تُخزن عناوين الإنترنت للمزوّدين الخاصين بك، تماماً مثلما تخزّن معلومات الاتصال الأخرى الخاصة بهم، واعرض العناوين في نموذج مزودين. عندما تحرك مؤشر الماوس فوق مربع النص الذي يعرض عنوان الارتباط التشعبي، يتغير رمز المؤشر إلى إصبع إشارة ويعرض شريط المعلومات عنوان الارتباط التشعبي. يأخذك نقر الارتباط التشعبي إلى هدف الارتباط التشعبي.

يُمكّنك نوع بيانات Hyperlink من تخزين المعلومات في ثلاثة أجزاء. الجزء الأول، *displaytext* يعد النص الذي تريد أن تعرضه في الحقل، أما الجزء الثاني، *address* يعد عنوان الارتباط التشعبي، والجزء الثالث، *subaddress* يعد العنوان الفرعي للارتباط التشعبي. يعتبر الجزء الأول اختياري. يتم فصل الأجزاء عن طريق علامة الجنية، كما في *display-text#address#subaddress*. إذا أدخلت عرض النص في حقل Hyperlink، يوّضح أكسس فقط *displaytext* في الخلية ولا تعرض باقي العنوان، وإذا قمت بإلغاء *displaytext*، يوّضح أكسس فقط عنوان الارتباط التشعبي.

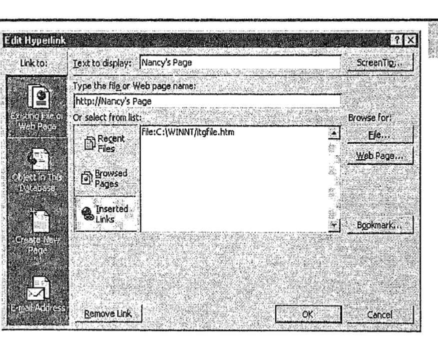
تُعد أسهل طريقة لإدخال هدف الارتباط التشعبي هو استخدام زر شريط أدوات Insert Hyperlink. على سبيل المثال، سنضيف حقل الارتباط التشعبي إلى جدول Employee. سيخزّن هذا الحقل URL لكل صفحة رئيسية خاصة بالموظف.

١- افتح جدول الموظفين في عرض Design. انقر في خلية Field Name الفارغة وقم بإدخال Home Page. اختر Hyperlink من مربع تحرير وسرد Datatype.

٢- احفظ الجدول وبدّل عرض Datasheet.



٣- انقر خلية Home Page للوظيفة الأولى واكتب النص الذي تريد عرضه. على سبيل المثال، اكتب **Nancy's Page** وبعد ذلك انقر زر Insert Hyperlink في شريط الأدوات. يمكن أن تحدد ملف لترتبطه به أو تحدد اسم صفحة ويب من قائمة لملفات قد تم التوصل إليها مؤخراً أو تُحدد الروابط المُدرجة الموجودة. "راجع الشكل ١-٢٢". افتراضياً، إذا قمت بإدخال الملف أو اسم صفحة ويب في مربع النص بالأسفل الذي يظهر البروتوكول //:http الذي يتبعه نص العرض. ستحتاج إلى مسح مربع النص قبل إدخال عنوان الارتباط التشعبي.



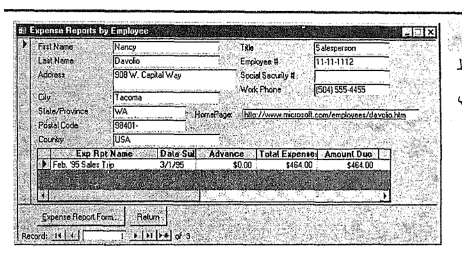
الشكل ١-٢٢ استخدم حوار Edit Hyperlink لإدخال عنوان وعنوان فرعي.

٤- قم بإلغاء النص في مربع النص Web Page Name أو File Type والنوع في URL مثل http://www.microsoft.com/employees/davolio.htm زائف للصفحة الرئيسية. الموظف الأول، انقر OK. يعرض أكسس نص الارتباط التشعبي في الطريق القياسي: يظهر النص وقد تم تسطيره ولونه الأزرق.

٥- انقر سجل آخر لتحفظ الإدخال وتعيد تعيين المؤشر. عندما تُحرّك المؤشر للوراء فوق الخلية، يتغير رمز المؤشر إلى اليد بإصبع يشير إلى الأشياء. إذا كنا قد أدخلنا URL حقيقي، فإن نقر الخلية سيفتح المستعرض ويعرض المستند.

٦- لتقوم بتحرير الارتباط التشعبي، انقر يمين الخلية واختر Hyperlink من قائمة الاختصار. انقر في مربع Display Text في القائمة الخارجية السريعة. حدد النص "Nancy's Page"، واضغط Delete لتلغي عرض النص، وبعد ذلك اضغط Enter. ستوضح الخلية الآن عنوان الارتباط التشعبي. لاحظ أنك تستطيع تغيير العنوان أو العنوان الفرعي للارتباط التشعبي عن طريق اختيار Edit Hyperlink من القائمة الخارجية السريعة لتعرض حوار Edit Hyperlink.

٧- افتح Expense Reports عن طريق نموذج Employee في عرض Design. اعرض قائمة Field "باختيار View ⇦ Field List" واسحب حقل Home Page إلى النموذج الموجود أسفل مربع النص Work Phone. قم بالتبديل إلى عرض Form. لقد تم عرض الارتباط التشعبي الخاص بالصفحة الرئيسية للموظف في مربع نص Home Page "راجع الشكل ٢٣-١".



الشكل ١-٢٣ عرض ارتباط تشعبي تم طيّه في نموذج.

استخدام ارتباط تشعبي كعنصر تحكم نموذج لم يتم طيه

عندما لا تريد أن يتغير الارتباط التشعبي مع كل سجل، يمكن أن تستخدم عنوان، زر أمر، أو صورة لتقوم بإنشاء ارتباط تشعبي على نموذج. على سبيل المثال، عندما تريد أن تستخدم ارتباط تشعبي لتنتقل إلى نموذج آخر أو إلى تقرير، قم بإنشاء الارتباط التشعبي مباشرة على النموذج. ولكي تُنشئ ارتباط تشعبي لم يتم طيّه، ضع عنوان، زر أمر، أو صورة على النموذج. يمكن أن تستخدم خاصية Caption للعنوان أو لزر الأمر حتى تصف الارتباط التشعبي. ولكي تقوم بإنشاء الارتباط التشعبي، عيّن خصائص عناصر تحكم Hyperlink Address، Hyperlink SubAddress. لكي تُنشئ ارتباط تشعبي لكائن آخر في قاعدة بيانات أكسس الحالية، اترك خاصية Hyperlink Address فارغة.

لكي تستكشف الارتباطات التشعبية التي لم يتم طيّها، سنقوم بإنشاء ثلاث أنواع:

- ارتباط تشعبي لعنوان يفتح ويعرض كائن آخر في نفس قاعدة البيانات.
- ارتباط تشعبي لصورة منقورة تفتح وتعرض كائن في قاعدة بيانات أكسس أخرى.
- ارتباط تشعبي لزر أمر يفتح المستعرض ويعرض صفحة ويب على World Wide Web.

إنشاء ارتباط تشعبي له عنوان

لكي تقوم بإنشاء ارتباط تشعبي كعنصر تحكم للعنوان، اتبع هذه الخطوات:

١- قم بإنشاء نموذج لم يتم طيّه جديد يسمى frmHyperlinks. افتح خصائص النموذج بالنقر على الكتلة المربعة في أعلى يسار ركن إطار النموذج، بعد ذلك انقر زر الخصائص. عيّن خاصية Caption إلى Hyperlink.

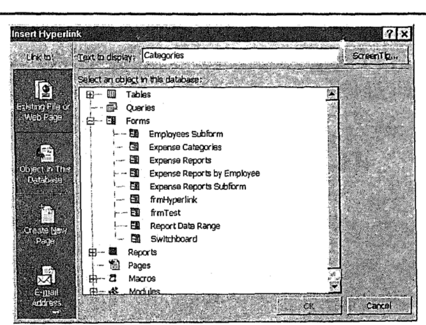
ملاحظة يجب أن يتم حفظ النموذج الجديد لكي يتم تسميتها.

٢- ضع عنصر تحكم العنوان على نموذج ونوع Categories مباشرة في ورقة خاصية العنوان. انقر زر Build على يمين مربع الخاصية لتعرض حوار Insert Hyperlink.

٣- انقر الكائن في زر Database هذا. يعرض حوار Insert Hyperlink الكائنات في قاعدة البيانات الحالية المُدرجة في تنسيق الشجرة حسب نوعها "راجع الشكل ١-٢٤".

٤- اختر نموذج Expense Categories وانقر OK. تعيّن أكسس خاصية Hyperlink SubAddress إلى Form Expense Categories وقم بإنشاء الارتباط التشعبي واعرض تعليق العنوان بالأزرق للنص الذي قد تم تسطيره.

٥- احفظ النموذج، قم بالتبديل إلى عرض Form وانقر العنوان حتى يفتح نموذج Expense Categories.



الشكل ١-٢٤
استخدم حوار Insert Hyperlink لتختار كائن قاعدة البيانات.

إنشاء صورة تم النقر عليها

توجد هنا خطوات لإنشاء صورة منقورة:

١- افتح نموذج Hyperlinks في عرض Design واختار Insert ⇦ Picture. حدد صورة في حوار Insert Picture. وسنستخدم ملف dove.wmf في مجلد Microsoft Office\Clipart\Popular. ولكي تُعيد تحجيم الصورة، عيّن خاصية Size Mode إلى Stretch، وانقر ركن لمستطيل تحديد الصورة، واسحب إلى الحجم المطلوب. سنستخدم هذه الصورة لإنشاء ارتباط تشعبي لنموذج في قاعدة بيانات أكسس أخرى.

٢- انقر زر Properties مع الصورة المحددة. انقر زر Build المجاورة لخاصية العنوان الفرعي Hyperlink لتعرض حوار Insert Hyperlink. انقر Object في زر Database لترى قائمة الكائنات في قاعدة البيانات "راجع الشكل ١-٢٤". حدد نموذج Expense Categories وانقر Ok. تستند أكسس خاصية Hyperlink SubAddress إلى Form Expense Categories.

٣- احفظ النموذج، قم بالتبديل إلى عرض Form، وانقر الصورة يتم تصغير النموذج الحالي ويتم عرض نموذج Expense Categories.

إنشاء ارتباط تشعبي لزر أمر

تقوم الخطوات الآتية بإنشاء زر أمر كارتباط تشعبي:

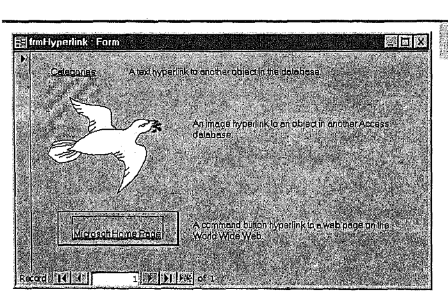
١- قم بالتبديل إلى عرض Design. وإذا كان ضرورياً انقر زر Control Wizards لكي لا تقوم بتحديد المعالج "يتم تنشيط Control Wizards عندما يتم الضغط على الـزر"،

بعد ذلك انقر أداة Command Button وارسم زر أمر على النموذج. ستستخدم هـذا الزر لإنشاء ارتباط تشعبي لصفحة ويب على World Wide Web.

٢- قم بإسناد خاصية Caption إلى Microsoft Home Page.

٣- انقر خاصية Hyperlink Address في ورقة خاصية زر الأمر، وأدخل URL إلى الصفحة الرئيسية الخاصة بمايكروسوفت: http://www.microsoft.com. فعندما تضغط على Enter تُنشئ أكسس الارتباط التشعبي، وتُغيّر لون خط الزر إلى أزرق، وتقوم بتسطير النص لتشير إلى ارتباط تشعبي.

٤- احفظ النموذج وبدّله إلى عرض Form "راجع الشكل ١-٢٥". انقر زر الأمر. إذا كان اتصال Internet الخاص بك مفتوح ويكون عندك Internet Explorer مثبّتة، يبدأ مثال جديد خاص Internet Explorer ويجدد المستعرض مكان الصفحة الرئيسية الخاصة بمايكروسوفت ويعرضها.



الشكل ١-٢٥ نموذج Hyperlink مع ارتباطات تشعبية لعنوان ما، لصورة، ولزر أمر.

تغيير نوع عنصر تحكم الارتباط التشعبي

يمكن أن تغيّر نوع عنصر التحكم للارتباط التشعبي بعد أن تُنشئ عنصر تحكم ارتباط تشعبي كواحد من ثلاثة أنواع. ففي عرض Form Design، حدد عنصر التحكم واختار Format ⇦ Change. تعرض القائمة الخارجية السريعة أوامر لنوعي عنصري التحكم الآخرين كأوامر نشطة. انقر النوع الذي تريد أن تغيره إليه.

إذا غيّرت من عنوان أو زر أمر إلى صورة، تعرض خاصية Picture الخاصة بعنصر تحكم الصورة كلمة "name" لأنك لم تحدد صورة ما. ولكي تدرج صورة انقر زر Build على يمين خاصية Picture واختار صورة نقطية لتعرضها.

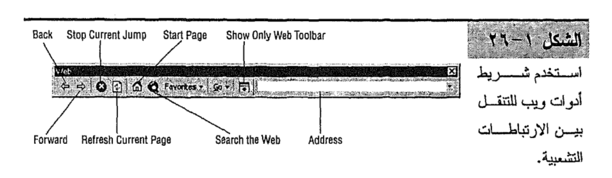
وإذا غيّرت من صورة إلى عنوان أو زر أمر، فتستند أكسس خاصية Caption الخاصة بعنصر التحكم الذي قد تم تغييره إلى قيمة خاصية Hyperlink. فإذا كانت خاصية Hyperlink Address فارغة فهي تغيّر خاصية Caption إلى قيمة خاصية Hyperlink SubAddress.

استخدام شريط أدوات ويب

عندما تنقر ارتباطات تشعبية لتقفز إلى كائنات أخرى، يحافظ أكسس ٢٠٠٠ على قائمة تاريخية داخلية لأهداف الارتباط التشعبي الذي قمت بزيارتها. يوّفر أكسس شريط أدوات ويب للتنقل "راجع الشكل ٢٦-١". استخدم زر Back لترجع إلى هدف الارتباط التشعبي السابق وزر Forword لتذهب إلى هدف الارتباط التشعبي التالي على القائمة التاريخية "يمكن لزر Forword أن يأخذك فقط إلى هدف قد قمت بالفعل بزيارته". وإذا غيّرت رأيك عقب تنشيط الارتباط التشعبي، يمكن أن تنقر زر Stop على شريط أدوات ويب لتوقف تتبعك للرابطة.

الشكل ١-٢٦

استخدم شريط أدوات ويب للتنقل بين الارتباطات التشعبية.



يُعد استخدام صور، عناوين، أزرار أمر مثل الارتباطات التشعبية طريق سهل لفتح وعرض كائن آخر. فعندما تنشئ الارتباط التشعبي عن طريق إعداد خصائص Hyperlink SubAddress ،Hyperlink Address ستكون محدود بفتح كائن والانتقال إلى مكان داخل الكائن: لا تستطيع أن تحدُد الإجراءات الإضافية. بينما عندما تنقر عنصر التحكم لكي تنشط الارتباط التشعبي يتعرف عنصر التحكم على حدث Click. وإذا أردت أن تأخذ أي إجراء آخر، مثل إخفاء النموذج الذي يحتوي على الارتباط التشعبي أو يتزامن مع نموذج أو تقرير مفتوح، يمكن أن تكتب برنامج وتطلب أن تشغل أكسس البرنامج عندما تنقر عنصر التحكم. سنناقش برمجة أحداث Click فيما بعد في هذا الكتاب.

استخدام معالج مربع التمرير والسرد

يعرض Expense Reports عن طريق نموذج Employee في قاعدة بيانات Expenses سجل لكل موظف. يتضمن المثال الخاص بنا ثلاث موظفين فقط كبيانات للنموذج. فمع القليل

من الموظفين، يُعد إيجاد سجل لموظف مُحدد أمر بسيط لاستعراض السجلات التي تستخدم أزرار التنقل بأسفل النموذج. ومع الكثير من الموظفين، تحتاج طريقة أكثر كفاءة لتجد موظف مُحـــدد. يوفر لك Combo Box Wizard الحل. يمكن أن تستخدم Combo Box Wizard لتنشئ مربـــع تحرير وسرد للبحث. تستطيع عن طريق مربع التحرير والسرد للبحث أن تحدد قيمة من القائمة، وتحدد وتعرض أكسس آلياً السجل المطابق.

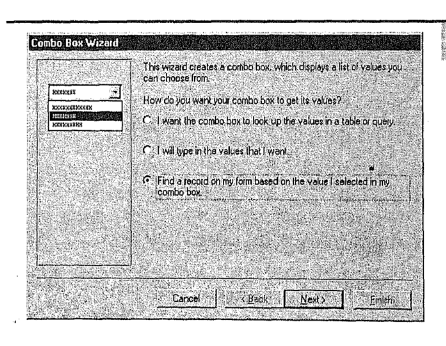
ففي مثالنا، سنستخدم المعالج لننشئ مربع تحرير وسرد الذي يضع كل اسم موظف بـــالترتيب الأبجدي، والإجراء الذي يتم تشغيله عندما تحدد اسم. يجد ويعرض الإجراء السجل للموظف الذي قمت بتحديده.



١-افتح Expense Reports عن طريق نموذج Employee في عـــرض Design. وإذا كان ضرورياً انقـــر زر Control Wizard لتنشـــط المعـــالجين. انقـــر آداة Combo Box في شريط الأدوات وبعد ذلك انقر النموذج المجاور لـــزر Return. تسأل الشاشة الأولى عن مصدر القيم التي تريدها في قائمة التحرير والسرد.

٢-اختر الاختيار الثالث لتُنشئ مربع تحرير وسرد للبحث "راجع الشكل ٢٧-١".

٣-تعرض الشاشة التالية قائمة للحقول في مصدر السجل الهام للنموذج "راجع الشكل ٢٨-١". وسنعرض فقط أسماء الموظفين في القائمة، ولكن ســـنحدد أيضـــاً حقـــل Employee لأنها تُعد المفتاح الأولي لجدول Employee. حـــــدد حقـــول First Name وlast Name وEmployee ID. بعد ذلك انقر Next. يُنشـــئ المعـــالج إجراء VBA يستخدم مفتاح Employee ID لتجد السجل الفريـــد المطـــابق لاســـم الموظف الذي قمت باختياره.

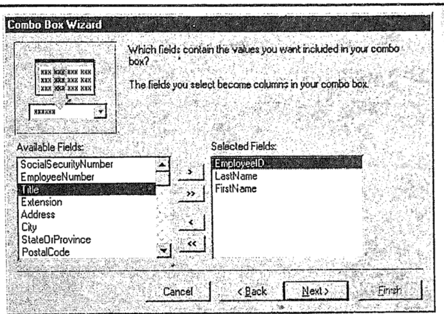


الشكل ١-٢٧

يمكن أن تســـتخدم Combo Box Wizard لتُنشـــئ مربع تحرير وسرد للبحث.

الشكل ١-٢٨

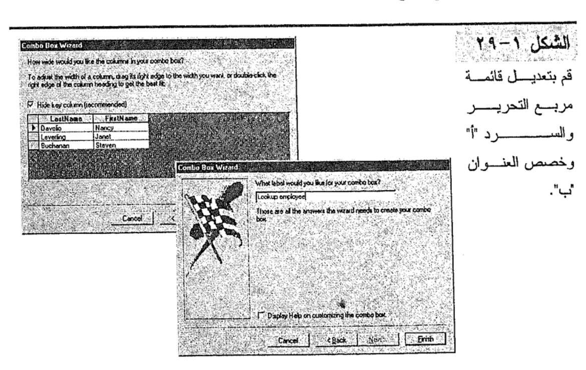
تُعد Available Fields حقول في النموذج الموجود في مصدر السجل.



٤- يمكن أن تقوم بتعديل شكل القائمة في الشاشة القادمة "راجع الشكل ١-٢٩ أ". تأكد من أن تترك مربع تدقيق عمود مفتاح Hide بعد التدقيق منه. انقر الزر Next. تعطيك الشاشة الأخيرة الفرصة لتخصيص العنوان "راجع الشكل ١-٢٩ ب". اكتب Lookup Employee في مربع النص.

الشكل ١-٢٩

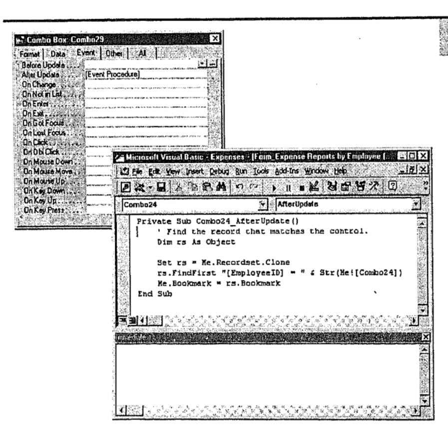
قم بتعديل قائمة مربع التحرير والسرد "أ" وخصص العنوان "ب".



٥- انقر زر Finish. يقوم المعالج بإنشاء مربع تحرير وسرد للبحث ويرفق إجراء VBA لإيجاد السّجل.

لاحظ أن ورقة الخاصية تشير إلى أنه يوجد إجراء حدث الآن لحدث After Update "راجع الشكل ١-٣٠ أ". تظهر أيضاً ورقة الخاصية أن المعالج قد قام بتعيين اسم افتراضي باستخدام بناء جملة Combo*nn*، حيث تُعد *nn* رقم معيّن تتبعي يتم التحكم فيه بواسطة أكسس ٢٠٠٠ الذي لا يساعد على الإطلاق في تعريف الغرض من مربع التحرير والسرد. "كان سيساعد بطريقة أفضل إذا كان Combo Box Wizard قد سمح لك بتسمية عنصر تحكم Cbo Employee بدلاً من ذلك" سنشرح كيف تعمل أحداث After Update و before Update في الفصل الثاني. أما الآن، لاحظ أن مربع التمرير والسرد يتعرف على حدث After Update في اللحظة التي تعقب تحديدك لاسم الموظف. انقر زر Build على يمين مربع خاصية After Update لتعرض إجراء VBA "راجع الشكل ١-٣٠ أ". يقوم هذا الإجراء *sunchronizes* النموذج إلى القيمة المعروضة في مربع التحرير والسرد عن طريق إيجاد وعرض السجل الذي يطابق مربع التحرير والسرد.

ولكي تخبر عمل المعالج، احفظ النموذج، قم بالتبديل للخلف إلى عرض Form، وحدد اسم الموظف من القائمة "راجع الشكل ١-٣١". ستجد أن النموذج يعرض آلياً السجّل المتزامن.



الشكل ١-٣٠

يقوم Combo Box Wizard بإنشاء إجراء حدث خاص بحدث "أ" After Update. ويتزامن إجراء الحدث "ب" مع سجل النموذج المسند إلى قيمة مربع التحرير والسرد.

الشكل ١-٣١

توفّر Expense Report المخصصة عن طريق نموذج Employee مسار إرجاع وبحث.

استخدام معالج لوحة التبديل

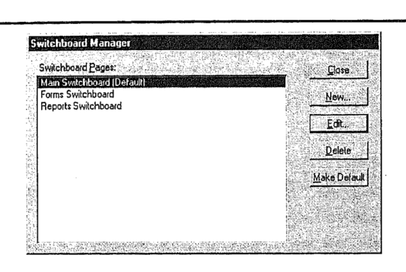
فعقب أن تُنشئ Database Wizard تطبيق رئيسي، يمكن أن تخصص عن طريق إضافة نماذج وتقارير. فعندما تضيف نموذج أو تقرير جديد، يمكن أن توّفر مسار له عن طريق إضافة زر أمر الذي يقوم بتشغيل برنامج أو عن طريق إضافة ارتباط تشعبي إلى النموذج المناسب أو إلى واحد من لوحات التبديل. ولكي تضيف زر إلى لوحة التبديل، انقر زر Change Switchboard Items على Main Switchboard. فسيقوم نقرك بتجميع مساعد أكسس آخر وهو Switchboard Manager.

ملاحظة

ففي أكسس، لا تستطيع أن تضع أزرار على جداول أو استعلامات لذا فلا يعرض عادةً الجداول والاستعلامات في واجهة مستخدم مخصصة. وبدلاً من ذلك، تقوم بإنشاء واجهة المستخدم الجديدة بأكملها خارج النماذج والتقارير. ولقد تم تصميم النموذج، صفحات تشغيل البيانات، وكائنات التقرير لكي تستجيب إلى زر النقر الخاص بالمستخدم، لتحديد قيمة من القائمة، أو إدخال قيمة في مربع النص.

يُعد الغرض من Switchboard Manager هو مساعدتك في تعديل نموذج Switchboard الذي تم إنشاءه عن طريق Database Wizard. يمكنك أيضاً أن تستخدم Switchboard Manager لتقوم بإنشاء Switchboard جديد إذا بدأت من لا شيء بدلاً من استخدام Database Wizard لتُنشئ قاعدة البيانات. يدرج بالقائمة Switchboard Manager "في الشكل ١-٣٢"

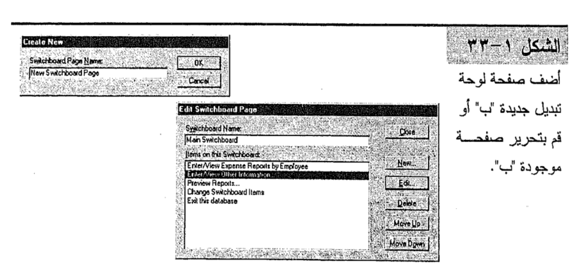
ثلاث صفحات للوحة التبديل في تطبيق Expenses. يظهر Main Switchboard كافتراضي، تُعد صفحة الافتراض واحدة يتم عرضها عندما تفتح أولاً نموذج Switchboard.



الشكل ١-٣٢ Switchboard Manager.

تعمل أزرار Switchboard Manager كالآتي:

- يقوم الزر الجديد بإنشاء صفحة لوحة تبديل جديدة. انقر زر New لتسمى صفحة لوحة التبديل الجديدة "راجع الشكل ١-٣٢ أ".
- يقوم زر Edit بتعديل الأزرار على صفحة لوحة التبديل المحددة. انقر زر Edit لتعرض حوار Edit Switchboard Page "راجع الشكل ١-٣٢ ب". يعرض الحوار العناوين للأزرار الموجودة على الصفحة بالنسبة للصفحة المحددة. يمكن أن تضيف زر جديد، وتُغيّر عنوان الزر الموجود، تلغي الزر، وتحرك الزر للأعلى وللأسفل في القائمة.
- يقوم زر Delete بإلغاء صفحة لوحة التبديل المحددة.
- يُغيّر زر Make Default الافتراض إلى صفحة لوحة تبديل محددة.



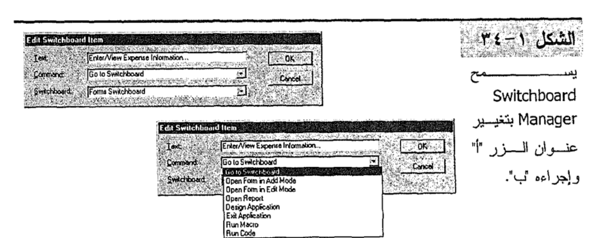
الشكل ١-٣٣ أضف صفحة لوحة تبديل جديدة "ب" أو قم بتحرير صفحة موجودة "ب".

وكمثال لاستخدام Switchboard Manager، ستُغيّر العنوان للـزر الثـاني علـى Main Switchboard إلى عنوان إخباري أكثر خاص Enter/View Expense Information.

١- انقر زر Edit وحدد البند الثاني بالقائمة عن طريق Main Switchboard المحددة فـي Switchboard Manager. انقر زر Edit على حـوار Edit Switchboard Page لتعرض حوار Edit Switchboard Item "راجع الشكل ١-٣٤ أ".

٢- قم بتغيير عنوان النص في المربع الأول إلى Enter/View Expense Information. يمكن أن تُغيّر إجراء الزار عن طريق تحديد أياً من الإجراءات الثمان في قائمة تحرير وسرد Command "راجع الشكل ١-٢٤ ب". فعقب أن تُحدد الإجراء، يتغير المربع الثالث ليعرض اختبارات مناسبة. ففي حالتنا، يأخذك أمر Go إلى Switchboard إلى صفحة لوحة تبديل أخرى، ويعرض مربع التحرير والسرد الثالث اسم صفحة لوحة التبديل "Forms Switchboard".

٣- انقر Ok لتغلق حوار Edit Switchboard Item، انقر Close لتغلق حوار Edit Switchboard Item، وانقر Close لتغلق Switchboard Manager. يعرض Main Switchboard العنوان الذي تم تغيره للزر الثاني.



الشكل ١-٣٤ يسمح Switchboard Manager بتغيير عنوان الزر "أ" وإجراءه "ب".

التحكم في واجهة المستخدم

فعقب أن تستخدم Switchboard Manager و Control Wizard يجب أن يكون تطبيقك المخصص أسهل في استخدامه، ولكن ما زلت لا تريد أن تقلبه إلى مبتدئ. ففي هذه المرحلة، تكون قد صممت مسارات تنقل ولكن لا تفيد المستخدم تلك المسارات. تعرض بيئة الأمر المضمنة الكاملة لأن جميع أوامر القائمة وأوامر قائمة الاختصار، أزرار شريط الأدوات واختصارات لوحة المفاتيح المضمنة في أكسس تكون متوفرة للتبديل إلى عرض Design وتغيير

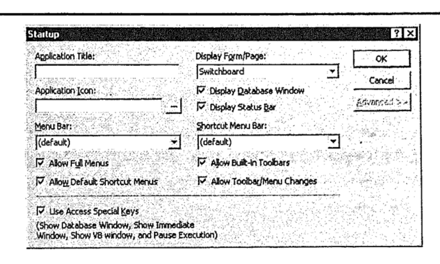
تصميم التطبيق. يتم عرض إطار Database أيضاً، على الرغم من تصغيره. يقوم المستخدم المبتدئ بالتشغيل الفوري للكائنات التي تُعد التطبيق الخاص بك.

وقبل أن تحوّل التطبيق إلى مبتدئ، يجب أن تحمي التطبيق عن طريق إخفاء أي أوامر يمكن استخدامها لتغيير تصميم التطبيق وبإخفاء إطار Database. يوّفر أكسس ٢٠٠٠ الأدوات للتحكم في واجهة المستخدم وحماية تطبيقك. ففي هذا المقطع، ستعرف ثلاث أدوات لا تتطلب البرمجة:

- يمكن أن تتحكم في العديد من ميزات بيئة أكسس عن طريق إعداد خصائص بدء التشغيل، على سبيل المثال، تستطيع تحديد هل تم عرض إطار Database أم لا عندما تبدأ تشغيل التطبيق، ويمكن أن تقوم بتعطيل اختصارات لوحة المفاتيح التي تعرض إطار Database " تركيبة المفاتيح Alt+F1 أو F11". بينما لا تستطيع بدون برمجة أن تمنع المستخدمين من الضغط على مفتاح Shift لتقوم بتمرير إعدادات خاصية بدء التشغيل الخاصة بك عندما تفتح قاعدة بيانات.
- يمكن أن تمنع إدخال لم يتم إجازته إلى قاعدة بيانات عن طريق تعريف كلمة المرور. يقوم التأمين البسيط لكلمة المرور بتقييد الإدخال داخل قاعدة بياناتك ولكن لا تمنح أي حماية بعد أن يفتح المستخدم الذي يعرف كلمة المرور قاعدة البيانات.
- يمكن أن تستبدل أشرطة الأدوات، القوائم، وأشرطة القوائم المضمنة بإصدارات مُخصصة تتضمن فقط الأوامر المطلوبة لتستخدم تطبيقك. يمكن بدون برمجة أن تقوم بتضمين الأوامر الخاصة بفتح كائنات قاعدة البيانات والأوامر المُضمنة أوامر مُخصصة لتشغيل مجموعة من الإرشادات ولكي تُخصص لوحة المفاتيح.

إعداد خصائص بدء التشغيل

تُملئ خصائص Startup كيف يبدو تطبيق قاعدة بيانات عندما يبدأ التشغيل. اختر Tools ⇦ Startup لتعرض حوار Startup وانقر Advanced لتعرض الاختيارات "راجع الشكل ١-٣٥".



الشكل ١-٣٥ يمكن أن تُعيّن معظم خصائص بدء التشغيل لقاعدة بيانات مستخدماً حوار Startup.

استخدم حوار Startup لتعيّن الخصائص الواضحة في الجدول ١-٣. يعد عنوان التطبيق وإعدادات الرمز فعالة بمجرد أن تختار حوار Startup. وتعد الإعدادات الأخرى فعّالة في المرة القادمة التي تفتح فيها قاعدة البيانات.

الجدول ١-٣: Startup Dialog في Startup Properties

خاصية بدء التشغيل	المواصفات
Application Title	تحدد النص الذي يظهر في شريط عنوان التطبيق. عندما لا يتم تعيين هذه الخاصية، تظهر كلمة "Microsoft Access" افتراضياً.
Application Icon	يحدد الرمز الذي يظهر في شريط عنوان التطبيق. عندما لا يتم تعيين تلك الخاصية، يظهر رمز مفتاح أكسس افتراضياً.
Display Form/Page	يحدد اسم النموذج أو الصفحة التي تعرض عن بدء التشغيل. عندما تُنشئ تطبيق مع Database Wizard، تستند هذه الخاصية إلى نموذج Switchboard افتراضياً.
Display Database Wizard	يحدد هل تم عرض Database Window أم لا، عندما تبدأ في تشغيل التطبيق.
Display Status Bar	يحدد هل تم عرض شريط المعلومات أم لا عندما تبدأ تشغيل التطبيق. طبيعياً، تريد أن تحافظ على الافتراض لتعرض شريط المعلومات لأنك تستخدمه لتعرض تعليمات على الشاشة.

الجدول ١-٣: Startup Dialog في Startup Properties

خاصية بدء التشغيل	المواصفات
Menu Bar	يعرض شريط قائمة اختصار مخصصة التي تستبدل شريط القائمة المضمنة في كل الإطارات الخاصة بتطبيقك ما عدا المكان الذي أنشأت فيه شريط قائمة مخصص للنموذج أو تقرير محدد.
Shortcut Menu Bar	يعرض شريط قائمة اختصار مخصص يستبدل شريط قائمة الاختصار المضمنة في كل الإطارات الخاصة بتطبيقك ما عدا المكان الذي أنشأت فيه شريط قائمة اختصار مخصص لنموذج، تقرير محدد، أو عنصر تحكم نموذج.
Allow Full Menus	يحدد إذا كان يجب على أكسس أن يعرض المجموعة الكاملة للقوائم المضمنة وأوامر القائمة أو فقط المجموعة المصغرة للقوائم المضمنة. لا تشمل المجموعة المصغرة أوامر تسمح لك بتعديل التطبيق.
Allow Default Shortcut Menus	يحدد إذا كان يجب على أكسس أن يعرض قوائم الاختصار المضمنة أم لا.
Allow Built-In Toolbar	يحدد إذا كان يجب على أكسس أن يعرض أشرطة الأدوات المُضمنة أم لا. يمكن أن تسمح مربع التدفق لإخفاء جميع أشرطة الأدوات المُضمنة ولكن ما زالت تعرض أشرطة الأدوات المُخصصة.
Allow Toolbar/Menu Changes	يحدد إذا كان يجب أن تسمح للمستخدمين أن يغيروا أشرطة الأدوات وأشرطة القوائم أم لا. فإذا قمت بمسح مربع التدفق، فذلك تعطل زر الماوس الأيمن، زر Close على أشرطة الأدوات وأمر Toolbars على قائمة View. يمكن أن تستمر في تحريك، تحجيم، وترس أشرطة الأدوات وشريط القائمة وتُخفي أولاً أشرطة الأدوات.

الجدول ١-٣: Startup Dialog في Startup Properties

خاصية بدء التشغيل	المواصفات
Use Access Special Keys	يحدد إذا كان يجب على أكسس أن يمكن لتركيبة المفاتيح المضمنة الخاصة أم لا. Alt+F1 أو F11 "تعرض، لا تُخفي، وتستعيد إطار Database "Ctrl+G "يعرض إطار Immediate"، Alt+F11 "تعرض VB Editor"، Ctrl+F11 "يبدل بين شريط قائمة مضمن وشريط قائمة مخصص"، Ctrl+Break "في قاعدة بيانات أو في مشروع، تعلّق تنفيذ إجراء VBA في مشروع، وتوقف أيضاً أكسس من استرداد السجلات من الخادم".

تُعد أسهل طريقة للتحكم في الأوامر المتاحة للمستخدم هي تعيين خصائص بدء التشغيل كما يلي:

- إخفاء إطار Database وتمنع المستخدم من عرض الإطار بعد بدء قاعدة البيانات.
- إخفاء جميع أشرطة الأدوات وقوائم الاختصار.
- استبدال القوائم المضمنة بمجموعة مصغرة بديلة من القوائم المضمنة.

تشمل المجموعة المصغرة فقط الأوامر المفيدة عندما يعمل المستخدم مع تطبيق قاعدة بيانات مُخصصة في العروض بدلاً من عرض Design وتقليل قوائم القائمة لتغيير تصميم كائن قاعدة بيانات بالإضافة إلى العديد من أوامر القائمة الأخرى. لا يشمل شريط القائمة المُصغرة لعرض Form. لا يشمل شريط القائمة قائمة View أو قائمة Tools ويقلل بعض أوامر القائمة على القوائم المتبقية.

سنستخدم حوار Startup لتخصيص شريط العنوان وحماية التطبيق.

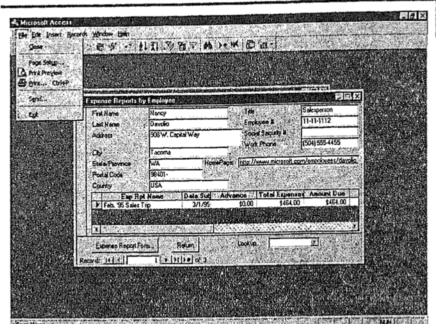
١- اختر Tools ⇦ Startup وانقر زر Advanced.

٢- قم بإسناد خاصية Application Title إلى Expenses.

٣- قم بمسح جميع مربعات التدقيق ما عدا المربع الذي يعرض شريط المعلومات "راجع الشكل ١-٣٧".

الشكل ١-٣٦

عرض Form المصغّر المصمم في شريط القائمة.



الشكل ١-٣٧

استخدام إعدادات Startup لتخفي إطار Database وتقيّد القائمة شريط الأدوات، وأوامر لوحة المفاتيح المتوفرة.

Startup
Application Title:
Expenses
Application Icon:
Menu Bar:
(default)
Allow Full Menus
Allow Default Shortcut Menus
Use Access Special Keys
(Show Database Window, Show Immediate Window, Show VB window, and Pause Execution)
Display Form/Page:
Switchboard
Display Database Window
Display Status Bar
Shortcut Menu Bar:
(default)
Allow Built-in Toolbars
Allow Toolbar/Menu Changes
OK
Cancel
Advanced >>

٤- انقر OK. لاحظ أن شريط عنوان التطبيق يتغيّر إلى Expenses فورياً. ففي إنشاء قاعدة بيانات Expenses يُنشئ Database Wizard إجراء يتم تشغيله عندما يفتح أولاً نموذج Switchboard ويشمل عبارات تُعيد عرض وتصغير إطار Database. يتم تشغيل الإجراء بعد أن تبدأ أكسس تشغيل قاعدة البيانات طبقاً لإعدادات بدء التشغيل ولهذا فهي تتجاوز إعدادات بدء التشغيل لتخفي إطار Database.

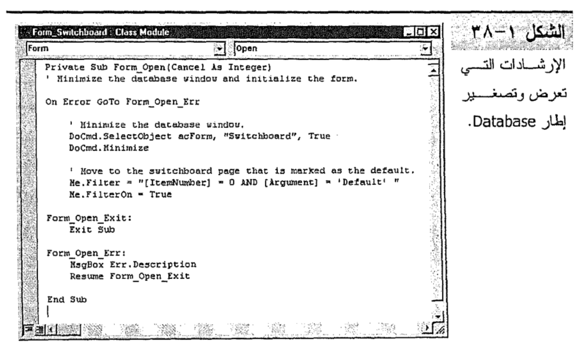
٥- افتح نموذج Switchboard في عرض Design. افتح ورقة الخاصية عن طريق تحديد View ⇦ Properties. "يمكن أيضاً أن تفتح ورقة الخاصية عن طريق النقر المزدوج لعنصر التحكم المربع إلى يسار المسطرة في شبكة التصميم". انقر خاصية Onopen

وبعد ذلك انقر زر Build إلى يمين خاصية المربع. يعرض إطار Module إجراء الحدث الذي يتم تشغيله عندما يتم فتح النموذج "راجع الشكل ١-٣٨". يعد السطران اللذان يتبعان التعليق Minimize the Database Window الإرشادات التي تعرض وتصغر إطار Database.

٦- انقر على يسار السطر Minimize the Database Window، واسحب لتحدد الثلاثة أسطر وبعد ذلك اضغط Delete.

٧- احفظ التغييرات واغلق نموذج Switchboard.

٨- انقر مربع Close في إطار Database. أعيد فتح قاعدة البيانات عن طريق اختيار Expense في القائمة التي بها قواعد البيانات المفتوحة مؤخراً في قائمة File. يفتح تطبيق Expenses عن طريق القوائم المصغَّرة، بدون أشرطة أدوات، وبدون قوائم اختصار.



```
Form_Switchboard : Class Module
Form                                    Open

Private Sub Form_Open(Cancel As Integer)
' Minimize the database window and initialize the form.

On Error GoTo Form_Open_Err

    ' Minimize the database window.
    DoCmd.SelectObject acForm, "Switchboard", True
    DoCmd.Minimize

    ' Move to the switchboard page that is marked as the default.
    Me.Filter = "[ItemNumber] = 0 AND [Argument] = 'Default' "
    Me.FilterOn = True

Form_Open_Exit:
    Exit Sub

Form_Open_Err:
    MsgBox Err.Description
    Resume Form_Open_Exit

End Sub
```

الشكل ١-٣٨ الإرشادات التي تعرض وتصغير إطار Database.

نعف طريق تلك الإعدادات، لن تتمكن من عرض إطار Database بواسطة الضغط على Alt+F1 أو F11 بعد أن تبدأ قاعدة البيانات. لاحظ أنه بينما يمكن أن تمرر إعدادات خاصية بدء التشغيل عن طريق ضغط مفتاح Shift عندما تفتح أولاً قاعدة البيانات "في الفصل ١٤ ستعرف كيف تعطّل تمرير مفتاح Shift أيضاً."

حماية تطبيقك عن طريق كلمة المرور

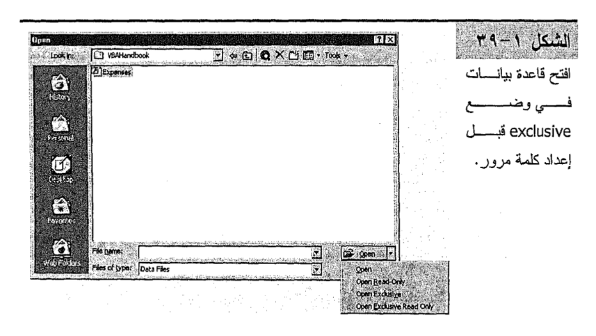
يمكن أن تمنع الأشخاص غير المسئولين من فتح تطبيقك عن طريق تعريف كلمة مرور. فعندما تحمي قاعدة بيانات بكلمة مرور، يجب أن يدخل المستخدمين كلمة المرور قبل أن يقوموا باستيراد كائناتهم داخل قاعدة بيانات أخرى أو تضغط قاعدة البيانات.

قبل أن تعيّن كلمة مرور، تحتاج أن تُنشئ تشغيل مقصور على قاعدة البيانات. يجب أن تغلق قاعدة البيانات قبل أن تحدد وضع مقصور. فعندما تفتح قاعدة بيانات في وضع exclusive، تمنع أي شخص آخر من فتح قاعدة البيانات.

ستعرف كلمة مرور لقاعدة بيانات Expenses:

١- اغلق قاعدة البيانات عن طريق نقر Exit لزر قاعدة البيانات تلك في Main Switchboard.

٢- اختر File ⇦ Open، حدد Expenses، وانقر Open Expenses في قائمة الإفلات لأسفل Open بينما تضغط على مفتاح Shift "راجع الشكل ١-٣٩".



الشكل ١-٣٩
افتح قاعدة بيانات في وضع exclusive قبل إعداد كلمة مرور.

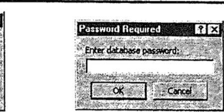
٣- اختر Tools ⇦ Security ⇦ Set Database Password. استخدم حوار Set Database Password لتعيين كلمة مرور "راجع الشكل ١-٤٠ أ".

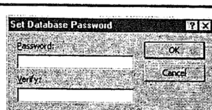
٤- اكتب **expenses** في مربع نص Password. تُعد كلمات المرور متحسسة لحالة الأحرف في أكسس. اكتب **expenses** مرة ثانية في مربع نص Verfiy. بعد ذلك انقر **OK**. انقر مربع Close في إطار Database.

٥- اختر Expenses من القائمة التي بها قواعد البيانات المفتوحة مؤخراً في قائمة File. أدخل **expenses** في حوار Password Required "راجع الشكل ١-٤٠ ب". يفتح قاعدة البيانات بمجموعة مصغرة من القوائم.

الشكل ١-٤٠

استخدم حوار Set Database Password "أ" لتعيّن كلمة مرور وحوار Password Required "ب" لتكسب الإدخال إلى قاعدة بيانات تحميها كلمة المرور.





٦- نحتاج التشغيل للقوائم الكاملة للمقطع القادم، حتى تستطيع أن نعين بدء قاعدة البيانات. انقر Exit لزر قاعدة البيانات. اضغط مفتاح Shift واختر Expenses من القائمة التي بها قواعد بيانات تم فتحها مؤخراً في قائمة File. أدخل كلمة المرور واضغط مفتاح Shift حينما تنقر OK. تفتح قاعدة البيانات بمجموعة كاملة من القوائم.



تأكد من أنك تسجل كلمة المرور في مكان ما. فإذا نسيت أو فقدت كلمة المرور الخاصة بك، لن تستطيع أن تفتح قاعدة البيانات.

على الرغم من أن إضافة كلمة مرور إلى قاعدة البيانات يُعد طريقة سهلة لتقييد الإدخال إلى قاعدة البيانات إلا أن حماية كلمة المرور محدودة جداً.

- يستطيع أي شخص معه محرر قرص أو برنامج أداة مساعدة يمكن أن تمتع هذا عن طريق تشفير قاعدة البيانات. ولكي تشفّر قاعدة بيانات، اغلق قاعدة البيانات واختر Tools ⇦ Security ⇦ Encrypt/Decrypt Database وانقر OK. ستعمل على إيجاد اسم لقاعدة البيانات المشفّرة قبل أن يظهر التشفير الفعلي.

- لكي تزيل حماية كلمة المـــرور، اخـــتر ببســـاطة Tools ⇦ Security ⇦ unset ⇦ Database Password. أدخـــل كلمـــة المـــرور فـــي حـــوار unset Database Password وانقر OK. هذا يعني أن أي شخص يعرف كلمة المرور ويمكن أن يصـــل إلى أمر unset Database Password يستطيع أن يغيّر أو يمسح كلمة المرور.

تلميح هناك طريقة أفضل لحماية قاعدة بياناتك وهي اســـتخدام تـــأمين مســـتوى المستخدم الذي يوّفره أكسس. فمع تأمين مستوى المستخدم، يمكن أن تعرّف المستخدمين ومجموعات من المستخدمين وتحدد مستويات مختلفة للوصـــول إلى بيانات وكائنات قاعدة البيانات. وللحصول على مزيد مـــن المعلومـــات عن تأمين مستوى المستخدم.

إنشاء أشرطة أدوات وقوائم مخصصة

لقد استخدمنا حوار Startup لنعرض مجموعة مُصغّرة لأشرطة القائمة وأوامر القائمة ولإخفـــاء جميع أشرطة الأدوات وقوائم الاختصار. على الرغم من أننا قد حققنا درجة من الحماية لقـــاعدة البيانات الخاصة بنا، إلا أننا ما زلنا لم نجعل بيئة الأمر مساعدة مثلما يجـــب أن تكـــون. تخفـــي أشرطة القوائم أوامرها، يجب أن يبحث المستخدم الجديد خلال القوائم المنسدلة لتعلـــم مـــا هـــي الأوامر المتوفرة وبعد ذلك تتذكر مكان الأوامر. بالإضافة إلى ذلك، مـــن الممكـــن أن لا توفّـــر أشرطة القوائم المصغرة مجموعة الأوامر التي يريدها المستخدم. تُعد أشـــرطة الأدوات طريقـــة أفضل لعرض الأوامر المتوفرة. فمن خلال أزرار شريط الأدوات المرئي في جميـــع الأوقـــات وscreen Tips المعروضة مثل مستعرض المستخدم من خلال الأزرار، يعد تطبيقك أسهل بكثير لتتعلمه وتستخدمه. تعد قوائم الاختصار أيضاً مفيدة، فمعظم المستخدمين معتادين على اســـتخدام زر الماوس الأيمن لعرض قائمة اختصار.

ففي أكسس ٢٠٠٠، يمكن أن تمرر أشرطة قوائم مضمنة، قوائم اختصار، وأشـــرطة أدوات، بالإضافة إلى إنشاء أشرطة قوائم مخصصة وأشرطة أدوات باستخدام حوار Customize. يطلق أكسس ٢٠٠٠ على الثلاث كائنات اسم Command bars ويوّفر طرق مشتركة لتخصيصـــهم. على سبيل المثال، يمكن أن تصمم أشرطة القوائم التي تعرض قوائم بأزرار أمر ومربعات تحرير وسرد، وأشرطة أدوات تعرض أسماء قوائم لها قوائم منسدلة لديك مرونـــة كبـــيرة فـــي كيفيـــة تصميمك لأشرطة القوائم، قوائم الاختصار، وأشرطة الأدوات في أكسس ٢٠٠٠، ولكن يجـــب أن تتمسك بتصميمات Windows القياسية لتتجنب إثارة الحيرة عند المستخدمين.

لكي تعرض حوار Customize، انقر يميناً على شريط الأدوات واختر Customize من قائمة الاختصار. تُدرج علامة تبويب Toolbars بالقائمة كلاً من أشرطة الأدوات الجديدة، وأشرطة الأدوات المُضمنة التي يمكن أن تقوم بإنشائها "راجع الشكل ١-٤١ أ". يعني التدقق أمام بند ما أن البند قد تم عرضه حالياً. يمكن أن تعرض أي شريط أدوات عن طريق نقر مربع التدقق الخاص به. بالإضافة إلى أشرطة الأدوات، تشمل القائمة Menu Bar كبند تدقق يمثل شريط القائمة التي تم عرضها حالياً. وبما أن شريط القائمة التي تريد أن تعمل به قبل أن تفتح حوار Customize.

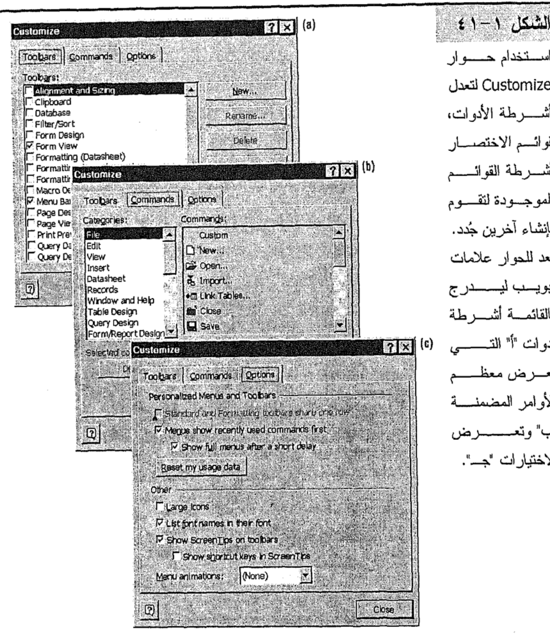
ملاحظة يمكن أن تحتاج أن تؤكد أن هذا الاختيار تم تعيينه في قائمة Startup، قبل أن تخصص شريط الأداة. ولكي تقوم بذلك، حدد Tools ⇦ Startup وتأكد من أن هناك تدقق في مربع تدقق Allow Toolbar/Menu Changes.

تعرض علامة تبويب الأمر في مربع القائمة على اليسار "راجع الشكل ١-٤١ ب". عندما تحدد فئة، يتغير مربع القائمة على اليمين ليعرض الأوامر في الفئة المحددة. تشمل أيضاً قائمة الفئات فئات لكل نوع من أنواع كائنات قاعدة البيانات ما عدا صفحات تشغيل البيانات والوحدات النمطية. إذا قمت بتحديد واحد من تلك الفئات، يعرض مربع القائمة على اليمين جميع الكائنات الخاصة بهذا النوع. على سبيل المثال، تحديد فئة ActiveX، التي تمثل جميع عناصر التحكم ActiveX التي تم تثبيتها وتسجيلها حالياً على الكمبيوتر الخاص بك، يعرض قائمة عناصر تحكم ActiveX الخاصة بك في مربع القائمة على اليمين.عندما تقوم بتحديد بند New Menu في قائمة Categories، تعرض قائمة الأوامر New Menu التي تستخدمها لتقوم بإنشاء قائمة جديدة. تسمح لك علامة تبويب Options بتحديد اختيارات شريط أمر إضافية "راجع الشكل ١-٤١ جـ". على سبيل المثال، يمكن أن تختار هل تقوم بعرض Screen Tips على أشوطة أدوات أم لا.

تشمل القائمة الموجودة في علامة تبويب Toolbars الخاصة بحوار Customize على بند Shortcut Menus الذي يمثل قوائم الاختصار. انقر بند Shortcut Menus لتعرض شريط قائمة له بنود قائمة لكل الفئات الأساسية بقوائم الاختصار "راجع الشكل ١-٤٢ أ". يمكن أن تنقر على بند قائمة تعرض قائمة منسدلة التي بها جميع قوائم الاختصار المتوفرة في هذه الفئة. انقر فئة Form لتعرض قوائم الاختصار المتوفرة عندما يكون النموذج إطار نشط "راجع الشكل ١-٤٢ ب". لاحظ أن هناك قوائم اختصار لكل عرض من عروض النموذج. أخيراً، يمكن أن تحدد Form View Record من القائمة لتعرض قائمة الاختصار الخاصة بهذا العرض "راجع الشكل ١-٤٢ ج".

تخصيص شريط أمر مضمن

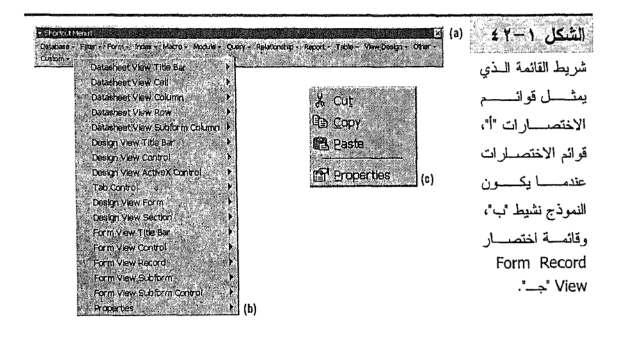
يجب أن تعرض شريط الأمر عندما يكون حوار Customize مفتوحاً لتخصص شريط أمر مضمن. وكما هو مفسر سابقاً، يمكن أن تعرض أي شريط أدوات أو أي قائمة اختصار عن طريق التدقق من البند في قائمة Toolbars، ولكن يمكن فقط أن تخصص شريط القائمة الذي تم عرضه قبل أن تفتح حوار Customize.



الشكل ١-٤١

استخدام حوار Customize لتعدل أشرطة الأدوات، قوائم الاختصار أشرطة القوائم الموجودة لتقوم بإنشاء آخرين جُدد. يعد للحوار علامات تبويب ليدرج بالقائمة أشرطة أدوات "أ" التي تعرض معظم الأوامر المضمنة "ب" وتعرض الاختيارات "جـ".

الشكل ١-٤٢ شريط القائمة الذي يمثل قوائم الاختصارات "أ"، قوائم الاختصارات عندما يكون النموذج نشيط "ب"، وقائمة اختصار Form Record View "جـ".



يمكن أن تخصص شريط الأمر المضمن الذي تم عرضه كالآتي:

- لتزيل أمر من أي شريط أمر، انقر زر الأمر أو الأمر واسحبه من شريط الأدوات وضعه داخل مساحة العمل.
- لتنقل أمر إلى مكان جديد على شريط أمر أو لتنقل أمر من شريط أمر واحد إلى شريط أمر آخر، انقر الأمر الذي تريده لتنقله وتسحبه إلى المكان الجديد الخاص به.
- لتنسخ أمر من شريط أمر تم عرضه إلى شريط أمر آخر تم عرضه، اضغط باستمرار على مفتاح Ctrl وأنت تسحب الأمر إلى مكانه الجديد.
- لتضيف أمر مضمن إلى شريط أمر يجب أن تنسخ الأمر من شريط أمر آخر أو تحدد الأمر في قائمة Command وتسحب الأمر إلى مكانه الجديد على شريط أمر.
- لتضيف أمر كي تفتح جدول، استعلام، نموذج، أو تقرير، اختر النوع الموجود في قائمة Categories، حدد الكائن في قائمة Command، واسحب الكائن إلى مكانه الجديد على شريط الأمر. ولتضيف أمر يقوم بتشغيل برنامج قمت بإنشائه كماكرو، اختر All Macros من قائمة Categories، حدد الماكرو الذي تريد تشغيله، واسحب الماكرو إلى مكانه الجديد على شريط الأمر.
- لتراجع التغييرات التي حدثت بشريط الأمر، حدد شريط الأمر في قائمة Toolbars في حوار Customize وانقر زر Reset. يعرض أكسس رسالة ليسألك أن تقوم بتأكيد إعادة التعيين.

ولكي تقوم بتعديل زر شريط أدوات أو اسم قائمة، مظهر، أو خصائص، انقر يمين البند لتعرض قائمة اختصار "راجع الشكل ١-٤٣ أ". يمكن أن تقوم بعمل تحديدات للتخصيصات الآتية:

- تنسخ، تلصق، أو تُعدّل صورة الزر.
- تحدد الأمر كنص، كصورة، كنص وصورة، أو كنمط افتراضي.
- تضيف شريط أفقي إلى قائمة أو شريط رأسي إلى شريط أدوات لتفصل الأوامر إلى مجموعات.
- تعيّن ارتباط تشعبي.

ولتعرض الخصائص للأمر، انقر أمر Properties لتعرض حوار Control Properties للبند المحدد. على سبيل المثال، يوّضح الشكل ١-٤٣ ب الحوار لأمر Auto Format الموجود في إطار Format. ولكي تقوم بتشغيل إجراء VBA الذي قمت بإنشائه كإجراء دالة "ستعرف أنواع الإجراءات المختلفة في الفصول القادمة"، تُدخل اسم إجراء الدالة في مربع on Action مستخدماً بناء الجملة ()=*functionname*.

الشكل ١-٤٣

يمكن أن تعدل اسم قائمة أو أوامر. انقر يميناً الأمر عن طريق حوار Customize المفتوح ليعرض قائمة الاختصار "أ". وانقر أمر Properties لتحرر خصائص "ب".

إنشاء وعرض أشرطة قوائم مخصصة

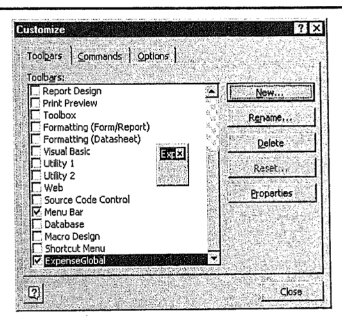
يمكن أن تقوم بإنشاء شريط قائمة مخصص يشمل أوامر القائمة المضمنة، وهي أوامر لفتح أي كائن إطار Window، والأوامر المخصصة الخاصة بك. يمكن أن تعرض شريط قائمة مخصص بطريقتين:

- إنشاء شريط قائمة مخصص فردي وإرفاقه إلى نموذج أو تقرير مُحدد. يتم عرض شريط القائمة عندما يكون النموذج أو التقرير كائن نشيط.
- إنشاء شريط قائمة عامة يتم عرضه في كل الإطارات ما عدا بعض النماذج والتقارير التي لها أشرطة القوائم المخصصة الفردية الخاصة بها.

وعلى سبيل المثال، سنقوم بإنشاء وعرض شريط قائمة عامة جديد. يوجد هناك العديد من الطرق لإنشاء قوائم وشريط القائمة. سننظر إلى القليل من التقنيات في تلك الخطوات.

١- اختر View ⇦ Toolbars ⇦ Customize أو انقر يميناً شريط القائمة وحدد Customize من قائمة الاختصار.

٢- انقر زر New واكتب Expense Global في حوار New Toolbar كاسم لشريط القائمة العام الجديد. انقر OK. تنشئ أكسس شريط أمر فارغ صغير، وأضيف بند Expense Global إلى قائمة أشرطة الأدوات" راجع الشكل ١-٤٤".



الشكل ١-٤٤
إنشاء شريط قائمة عام جديد.

٣- انقر زر Properties لتعرض حوار Toolbar Properties. استخدم هذا الحوار لتحدد إذا كان شريط الأمر هو شريط قائمة، شريط أدوات، أو قائمة اختصار أم "اختر Popup لقائمة اختصار". اختر Menu Bar في مربع تحرير سرد Type وانقر Close.

٤- اضغط استمرار مفتاح Ctrl وحدّد اسم قائمة File في شريط القائمة الذي تعرضه، اسحبه إلى الأمر الفارغ، وقم بإلغائه. عندما تسحب اسم قائمة، فإنك تسحب أيضاً قائمة pull-down menu.

٥- انقر قائمة File على شريط القائمة الجديد لتعرض قائمة pull-down menu. حدد خطوة بخطوة الأوامر واسحبهم من القائمة، اترك فقط أوامر Page Setup وSave As وClose وExit وPrint وPrint Preview.

٦- اضغط باستمرار مفتاح Ctrl وحدّد اسم قائمة Edit في شريط القائمة المعروض، اسحبه إلى يمين قائمة File في شريط الأمر الفارغ وقم بإلغائه. انقر قائمة Edit على شريط القائمة الجديد لتعرض قائمة pull-down menu. وحدد خطوة بخطوة الأوامر واسحبهم من القائمة، اترك فقط أوامر Paste وCopy وCut وUndo.

٧- حدد New Menu من قائمة Categories في علامة تبويب Command الخاصة بجوار Customize. انقر أمر New Menu في قائمة Commands واسحبها إلى يمين قائمة Menu على شريط القائمة الجديد. انقر يميناً في اسم القائمة "افتراضياً New Menu" وغيّر الاسم إلى Forms. انقر اسم القائمة الجديد لتعرض قائمة pull-down menu الفارغة الصغيرة.

٨- اختر فئة All Forms في علامة تبويب Commands الخاصة بجوار Customize، حدد Expense Reports عن طريق Employee، واسحبه إلى قائمة pull-down menu لقائمة Forms. حدّد Expense Categories واسحبهم إلى قائمة pull-down menu.

٩- انقر شريط أدوات Expense Global واسحبه ناحية أعلى الإطار لترسي شريط الأدوات الجديد أسفل أشرطة الأدوات الأخرى التي قد تعرضها.

١٠-حدّد علامة تبويب Toolbars وانقر زر Properties لتعرض حوار Toolbars Properties. يمكن أن تستخدم هذا الحوار لتسمح أو لتمنع تغيرات شريط القائمة. اختر Expense Global من قائمة التمرير والسرد وامسح جميع مربعات التدقق لتمنع التغيير أو التنقل. انقر Close وسيظهر حوار Customize. انقر Close مرة ثانية. ينتهي شريط القائمة العام.

١١-يمكن أن تعرض شريط القائمة العام عن طريق إعداد واحد من خصائص بدء التشغيل. اختر أمر Tools ⇦ Startup في قائمة Tools. انقر السهم الأسفل على مربع تحرير وسرد Menu Bar، حدّد شريط قائمة Expense Global وانقر OK. سيعرض شريط القائمة الجديد في المرة القادمة التي تبدأ فيها تشغيل التطبيق.

استخدام نفس التقنية لتنشئ شريط قائمة فردى لنموذج أو تقرير يعد الاختلاف الوحيد بين شريط القائمة بينما تحدد شريط قائمه ٤ عام كخاصية بدء التشغيل ، تحدد شريط قائمة فردى لنموذج أو تقرير عن طريق إسناد خاصية Menu Bar للنموذج أو التقرير إلى اسم شريط القائمة فعندما تعرض نموذج أو تقرير له شرط القائمة المخصص الخاص به، يعرض أكسس شريط

قائمة مخصص بدلا من شريط قائمة عام مخصص "أو شريط قائمه مضمن افتراضي إذا لم يكن لتطبيقك شريط قائمه عام".

إنشاء وعرض قائمة اختصار مخصصه

يمكن أن تنشئ قوائم اختصار مخصصه للنماذج ، التقرير ؛ وعناصر التحكم على النماذج ويمكن أن تنشئ قائمه اختصار فرديه لكل نموذج أو تقرير و قائمة اختصار عامه تعرض عندما لا يكون للنموذج أو التقرير النشط قائمه اختصار فردية.

وكمثال، سنقوم بإنشاء شريط قائمة اختصار مخصص بسيط سنستخدمه كقائمة اختصار عامه.

١- انقر يمين شريط القائمة وحدد نموذج CUSTOM من قائمه الاختصار.

٢- انقر زر New وقم بتسميه شريط الأمر الجديد Expense Shortcut انقر زر الخصائص في حوار Customize حدد Popup من مربع قائمه Type انقر زر Close سيختفي شريط الآمر الصغير ولكي تنشئ قائمه Type انقر OK وانقر زر Close سيختفي شريط الآمر الصغير ولكي تنشئ قائمه اختصار يجب أن تعرض شريط قائمه Shortcut Menu.

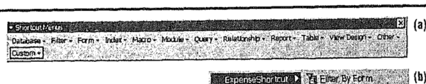
٣- انقر في بند Shortcut Menu في قائمه Toolbars أنقر السهم الأسود الصغير إلي يمين قائمه Customize يمين نهاية شريط قائمه Shortcut Menu لتعرض قائمه مفسر له تحتوي علي قائمة Expense Shortcut "راجع الشكل ٤٥-١". انقر السهم الأسود الصغير لتعرض قائمة انطلاق فأرغه صغيره حيث تستطيع إنشاء قائمة اختصار جديده سنقوم بإنشاء القائمة عن طريق سحب نسخ لأوامر التصفية من قائمة اختصار Form View.

٤- انقر قائمه Form علي شريط قائمه Shortcut Menu، انقر بند Form View Subform لتعرض قائمه الاختصار اضغط باستمرار مفتاح Ctrl وبعد ذلك حدد خطوة بخطوة أوامر apply Filter/Sort and Remove وFilter By Form واسحبهم إلى القائمة المخصصة Expense Shortcut "راجع الشكل ٤٥-١" اسحب إلى الأسهم السوداء الصغيرة لتعرض القوائم.

٥- انقر زر Close لتغلق شريط قائمة Shortcut Menu وحوار Customize.

الشكل ١-٥٤

قم بإنشاء قائمة اختصار مُخصصة.



٦- اختر Tools ⇦ Startup، حدد Expense Shortcut من مربع تحرير وسرد Shortcut Menu Bar وانقر OK. ففي المرة القادمة التي تبدأ فيها تشغيل التطبيق، يتم عرض قائمه الاختصار المخصصة عندما تنقر يمين نموذج ما.

استخدم نفس التقنية لتنشئ قائمه اختصار فرديه لنموذج تحكم أو تقرير بعد الاختلاف الوحيد بين قائمه اختصار فرديه وقائمة اختصار عامه هو كيفيه ترتيبك لعرض قائمة الاختصار حدد قائمة اختصار فرديه لنموذج، نموذج تحكم أو تقرير عن طريق إسناد خاصية Shortcut Menu Bar الخاصية بنموذج عنصر تحكم أو تقرير إلي اسم شريط قائمه الاختصار فعندما تعرض نموذج أو تقرير له قائمة اختصار مخصصه خاص به يعرض أكسس قائمة اختصار مخصصة بدلا من قائمه اختصار عامة مخصصة "أو القائمة المضمنة الافتراضية إذا لم يكن للتطبيق قائمة اختصار عامه".

إنشاء وعرض شريط أدوات مخصص

علي الرغم أنه من الممكن أن تعرض فقط شريط قائمه واحد وقائمه اختصار واحدة في وقت ما يمكن تعرض أي عدد من أشرطه الأدوات المخصصة والمضمنة هناك استراتيجية بسيطة لتوفير أشرطة أدوات مخصصة لتطبيقك وهي إخفاء جميع أشرطه الأدوات المضمنة "عن طريق مسح مربع تدفق Allow Built-In Toolbars في حوار Startup" اعرض أشرطة أدوات فردية مخصصة للنماذج والتقارير. يمكن أن تنفذ تلك الاستراتيجية البسيطة بدون برمجة.

وكمثال لذلك، سنقوم بإنشاء شريط أدوات عام بسيط.

١- انقر يمين شريط القائمة Customize وحدد من قائمه الاختصار.

٢- انقر زر New، قم بتسمية شريط الأدوات الجديد **ToolbarExpense** "راجع الشكل ١-٤٦ أ"، وانقر OK قم بإضافة أكسس إلى القائمة في علامة تبويب Toolbars.

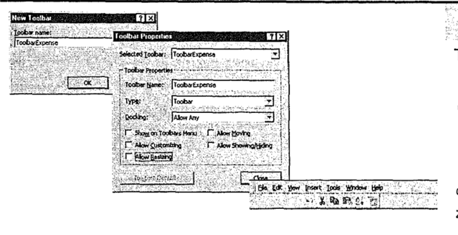
٣- فمن فئة Edit في علامة تبويب Commands الخاصة بجوار Customize حدد واسحب أوامر Undo وCut وCopy وPaste ومن فئة Records. اسحب أوامر Filter by Form وsort Ascending.

٤- انقر شريط العنوان لشريط الأدوات الجديد واسحب وقم بإرسائه أسفل شريط القائمة.

٥- انقر زر Properties علي علامة تبويب Toolbars الخاص Customize ليعرض حوار Toddler properties "راجع الشكل ١-٤٦ ب". استخدم هذا الحوار لتسمح أو تمنع تغير شريط القائمة اختر Toolbar Expense من قائمة تحرير وسرد وامسح مربعات التدفق لتمنع التخصيص وتعيد تحجيم أو تنقل شريط الأدوات الجديد.

٦- تأكد من أن Mecum Bar Toolbar Expense يعد أن بنود التدفق الوحيد في قائمه Toolbars في حوار Customize. انقر زر Close. يوضح الشكل ١-٤٦ ب شريط قائمة إطار Database المضمن وشريط الأدوات الجديد.

٧- انقر زر Close في إطار Database.



الشكل ١-٤٦ حوار Toolbar جديد لتسميه شريط أدوات مخصص "أ" وحوار Toolbar properties لإعداد خصائص "ب" الخاصة بشريط الأدوات الجديد "ج".

مراجعة تطبيق أكسس الآلي الخاص بنا

لقد قدم إليك هذا الفصل العديد من الأدوات التي يمكن أن تستخدمه لتجعل قاعدة البيانات آلية، لتقوم بتقييد القائمة وأوامر شريط الأدوات، ولتحمي تطبيقك بدون برمجة. لقد استكشفنا الأدوات عن طريق إنشاء تطبيق Expenses. دعنا ننظر إلى تطبيق Expenses ومراجعة عباراته الحالية.

١- اختر قائمة File ومن القائمة التي بها قواعد بيانات قد تم فتحها مؤخراً أسفل قائمة File، واختر Expenses لتفتح قاعدة البيانات. يعرض أكسس حوار Password Required.

٢- اكتب Expenses في مربع النص وانقر OK. يجب أن تستخدم جميع الأحرف الصغيرة، لأن كلمة المرور تُعد حساسة. "لن تقبل أكسس إصدارات أحرف مخلوطة، مثل Expenses". لقد تم إخفاء إطار Database وشريط القائمة العام وتم عرض شريط الأدوات المخصص.

٣- انقر Enter/View Expense Reports by Employee. انقر يميناً في النموذج لتعرض قائمة الاختصار العامة المخصصة.

٤- انقر زر Return وبعد ذلك انقر Exit لزر قاعدة البيانات تلك على Main Switchboard.

ولأن لقاعدة بيانات Expenses بعض الحماية من تغييرات التصميم، يمكن أن ترجعه بسهولة إلى شخص لا يعرف أكسس. خذ في اعتبارك هذا رغم أن إجراءات الحماية يمكن أن تزال معطلة بالضغط على Shift عندما تدخل كلمة المرور.

يمكن أن تستمر في استخدام التقنيات الموصوفة في هذا الفصل لتطور التطبيق. على سبيل المثال، عن طريق استخدام Command Button Wizard، يمكن أن تجعل من السهل التنقل من خلال مسارات ذو طريقتين إلى جميع نماذج وتقارير التطبيق. يمكن أيضاً أن تستخدم Command Button Wizard لتضيف مهمات آلية أخرى في الجدول ١-١. يمكن أن تضيف مربعات تحرير وسرد للبحث إضافية التي تستخدم Combo Box Wizard. يمكن أن تضيف نماذج وتقارير، ويمكن أن تستخدم Switchboard Manager لتقوم بتضمين الكائنات الجديدة على التبديل. يمكن أن تقوم بإنشاء أشرطة قوائم مُخصصة إضافية، قوائم اختصار، وأشرطة أدوات لتُسهل استخدام تطبيقك. وبالعمل فقط مع مساعي أكسس والحوارات المتنوعة للخاصية مثل حوار Startup وحوار Customize، لديك القوة لإنشاء تطبيقات قاعدة بيانات آلية تماماً بدون المزيد من الخوض في هذا الكتاب. هل تحتاج أن تقرأ المزيد؟ نعم. اقرأ المقطع القادم لتعرف لماذا.

خلف المعالجات والمساعدات

تُعد Database Wizard قوية جداً، ولكن يمكن أن تقوم فقط بإنشاء قواعد بيانات مستندة على القليل من القوالب التي تأتي مع أكسس. فإذا لم يقع تطبيقك داخل واحداً من تلك الفئات، فستحتاج أن تُنشئ قاعدة البيانات من لا شئ. وبعد ذلك ستستخدم المعالجات والمساعدات الآخرين لتجعل قاعدة البيانات آلية. وعلى الرغم أنه من المحتمل إنشاء تطبيق آلياً تماماً مع معالجي أكسس، فلا تزال التطبيقات التي تقوم بإنشائها محدودة في أنواع العمليات التي يمكن أن يقوموا بتنفيذها.

فعلى سبيل المثال، يقوم التطبيق الذي تم إنشائه مع المعالجين بتقييد التحقق من صحة البيانات لأنواع قواعد صحة البيانات التي تستطيع إدخالها في خصائص حقل الجدول وخصائص نموذج التحكم. هذا يعني أنك تستطيع استخدام فقط قاعدة فردية للتحقق من صحة الأشياء لتتحقق من صحة السجل، على الرغم أنه في بعض الحالات، يمكن أن تريد تنفيذ تتابع لاختبارات التحقق من

صحة الأشياء. أيضاً، إذا لم تكن البيانات التي تم إدخالها مرضية لقاعدة التحقق من صحة الأشياء، يمكن أن تعرض فقط رسالة فردية، رغم أنك قد تريد أن تعرض رسائل مختلفة القيمة التي تم إدخالها.

تتم اختبارات التحقق من صحة الأشياء وفقاً لقواعد الزمن الافتراضية. على سبيل المثال، تختبر أكسس قواعد التحقق من صحة الأشياء للتحكم في نموذج عندما تحاول أن تضع علامة التبويب خارج عنصر التحكم وتختبر قواعد التحقق من صحة الأشياء للسجل، مثل تفرد قيمة المفتاح الأول "تكامل الكيان"، عندما تحاول أن تحفظ السجل. غالباً، ستريد أن تُغيّر زمن اختبارات التحقق. على سبيل المثال، يمكن أن تريد اختبار التفرّد عندما tab out المفتاح الأول بدلاً من الانتظار إلى أن يتم إدخال البيانات الأخرى للسجل.

هناك مثل آخر لعملية قاعدة البيانات التي لا تستطيع إعدادها مع المعالجين وهو *transaction processing*. يُعد *transaction* هو مجموعة من العمليات التي تتناولها كوحدة فردية، إما أن تنفذ جميع العمليات في المجموعة أو لا تنفذ أياً منهم. إذا بدأت المعاملة وفشلت عملية واحدة، فسترد العمليات السابقة عن طريق إرجاع البيانات إلى الحالة التي كانت عليها قبلى أن تبدأ المعاملة. يعد المرشح الجيد لمعالجة المعاملة هو عملية أرشيف، تقوم فيها بإلحاق السجلات إلى الجداول التاريخية وبعد ذلك تقوم بإلغائهم من جداول البيانات الحالية. يجب أن تظهر كل العمليات أو لا يظهر أياً منهم. لا توّفر أكسس مساعد لمعالجة المعاملة.

بالإضافة إلى ذلك، يمكن أن تريد أن يكون تطبيق أكسس الخاص بك من المكونات في "mega application" المعقدة التي تتضمن جدول بيانات إكسل ومستندات وورد وتتطلب من أكسس، إكسل، وورد أن يتصلوا ببعضهم البعض آلياً وفقاً لإرشاداتك المحددة. أو يمكن أن تريد أن تستفيد بميزة مئات الإجراءات المفيدة في مكتبة الإجراءات التي يوفرها نظام تشغيل Windows أو الإجراءات المخزنة في مكتبات التطبيقات الأخرى "راجع الفصل ١٥ للحصول على المزيد من المعلومات عن مكتبات الإجراء". لا تستطيع أن تستخدم المساعدين للاتصال خارج أكسس.

يأخذك باقي هذا الكتاب خلف المساعدين. ستقوم بإنشاء برامجك الخاصة لتجعل قاعدة بيانات آلية من لا شئ باستخدام لغة برمجة Access VBA. ففي برمجة VBA تقوم بإنشاء إجراءات مخزنة إما في وحدات نمطية قياسية مدرجة بالقائمة في لوح Modules الخاص بإطار Database، أو في وحدات نمطية التقرير أو النموذج الذي تم إنشائها على شكل نماذج وتقارير. يمنحك أيضاً أكسس ٢٠٠٠ وحدة نمطية لفئة مستقلة.

يسمى جزء مايكروسوفت أكسس الذي قد تفاعلت معه لتُنشئ وتعرض كائنات قاعدة بيانات Access Application، ويسمى الجزء الذي يريد البيانات في قاعدة بيانات *database engine*. ففي قاعدة البيانات أكسس، يُعرف محرك قاعدة البيانات Microsoft Database Engine، أو MSDE. تستخدم الجزئين الخاصين بـ *data-access languages* لتتصل ببعضها البعض.

لم تعد لغات تشغيل البيانات لغات برمجة وحيدة، فبدلاً من ذلك، فهم لغات لهم غرض خاص تستخدمهم مع لغة البرمجة لتحدد البيانات التي تريد أن تسترد ما من أو تضيقها إلى الجداول.

لقد استخدمت واحدة من لغات تشغيل البيانات منذ بداية عملك مع أكسس، ربما بدون أن تدرك ذلك. تُعد Sructured Query Language أو SQL (تنطق في بعض الأوقات "Sequel") اللغة التي يستخدمها غالباً List Box Wizards وcombo Box وReport وForm ليقوموا بتحديد مصادر السجل ومصادر الصفوف. تستخدم برمجة VBA، SQL لتعمل مع الجدول والاستعلامات.

تسمى لغة تشغيل البيانات الآخر Data Access Objects أو DAO تستخدم لغة DAO فقط في برمجة VBA. تستخدم DAO لتكتب إرشادات لإنشاء ومعالجة الكائنات التي أدبرت عن طريق محرك قاعدة بيانات Jet. تشمل تلك الكائنات الجداول والاستعلامات في قاعدة بياناتك، ولكن تشمل أيضاً الكائنات التي استخدمها Jet ليقيد الوصول إلى البيانات وإدارة قاعدة البيانات. يوجد هناك اثنين من الأدوار الرئيسية خاصة DAO.

- لتوفير طريقة بديلة لتحديد البيانات "على الرغم أنه في معظم الحالات يعد أداء طريقة SQL أفضل بكثير."
- لتوفير القدرة على إنشاء وتعديل كائنات قاعدة البيانات كجزء من إجراءات VBA.

بالإضافة إلى DAO، ستعرف المزيد عن ActiveX Data Objects أو ADO. يقوم ADO بدوره كمترجم بين كائن Application وMSDE.

تقوم معظم الوقت بإنشاء جميع الجداول والاستعلامات في تطبيق باستخدام إطارات Design الخاصة بهم. بينما توجد مرات عندما فضلت فيها أن تُنشئ جداول واستعلامات برمجية. على سبيل المثال، استخدمت DAO عندما كتبت إجراء VBA الذي أنشأ جدول مؤقت جديد كجزء من عملية استيراد بيانات آلياً.

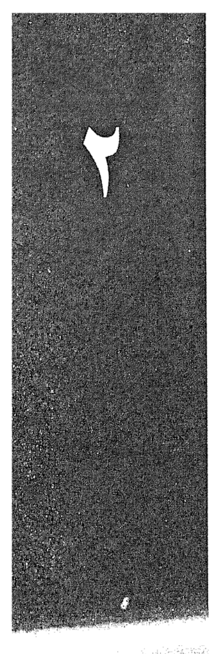
خلاصة

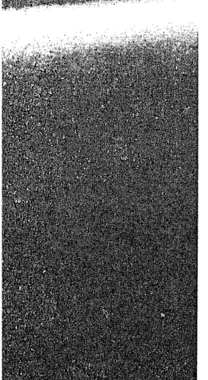
لقد قدم لك هذا الفصل الأدوات التي يمكن أن تستخدمها لتجعل قاعدة بيانات آلية بدون القيام بأي برمجة. تعتمد الأدوات على إعداد الخصائص في الحوارات وأوراق خاصية وعلى استخدام المعالجات والمساعدات لكتابة برامج لك. لقد وضح المعالجين مفهوم كتابة برنامج كأنه إما ماكرو أو إجراء VBA وبعد ذلك قاموا بترتيبات من أجل أكسس ليتم تشغيل البرنامج آلياً عندما يقوم المستخدم بعمل إجراء. يعد بيئة البرمجة في أكسس event driven حدث. تتسبب إجراءات مستخدمين معينة في جعل عناصر التحكم، النماذج، والتقارير يتعرفوا على التغييرات المسماه أحداث. يشغل البرامج عندما تظهر الأحداث.

ما يلي هو تلخيص للمساعدين الذين قد قمنا باستكشافهم:

- تقوم Database Wizard بإنشاء تطبيقات آلية جزئياً، وتكتمل بلوحات التبديل الخاصة بأزرار الأمر والتنقل لفتح وتزامن النماذج والتقارير ومربعات الحوار المُخصصة التي تجمع إدخال المستخدم قبل تحديد سجلات لتقارير التلخيص.
- يقوم Command Button Wizard بإنشاء أزرار أمر مع أشكال للخط لجعل التنوع الواسع لمهمات قاعدة البيانات البسيطة آلية.
- Command Box Wizard بإنشاء مربعات تحرير وسرد للبحث التي يمكن أن تبحث آلياً وتعرض سجل يطابق القيمة الموجودة في مربع التحرير والسرد.
- يقوم Switchboard Manager بإنشاء وتعديل نموذج Switchboard لتوفير مسارات تنقل للنماذج، التقارير ولوحات التبديل الأخرى.
- يسمح لك حوار Startup بالتحكم في كيف يبدأ تطبيقك أن يشتغل ويسمح لك بحماية تصميم تطبيقك.
- يسمح لك حوار Customize بإنشاء أشرطة قوائم متخصصة، قوائم اختصار، وأشرطة أدوات.

أنت الآن مستعد لاستخدام البرمجة لتوفر الاتصالات الآلية بين الكائنات في قاعدة بياناتك. ليس فقط كائنات إطار Database المعروف لك، ولكن أيضاً الكائنات الإضافية التي ستكون جديدة بالنسبة لك. يذكرك الفصل القادم بالتفكير في كيفية تصميم الكائنات في أكسس وفي كيفية تحكمك فيهم عن طريق برامجك.





البدء مع الكائنات والأحداث

يعرض هذا الفصل مفهومين أساسيين في برمجة VBA وهما الكائنات والأحداث. يناقش الجزء الأول من هذا الفصل كائنات إطار Database المعروفة وكائنات بيانات التشغيل غير المعروفة جيداً. تستخدم كائنات تشغيل البيانات بواسطة محرك قاعدة البيانات وهو الذي يدير البيانات ويتحكم فيمن يستطيع استخدام كائنات إطار Database وعندما يتم العمل تفاعلياً مع أكسس فأنت لا تعمل صراحة مع كائنات تشغيل البيانات بمحرك قاعدة البيانات هو الذي يعمل في الخلفية منشئاً وديراً بصورة تلقائية لكائنات تشغيل البيانات. على سبيل المثال، عندما تقوم بإنشاء كائن إطار Database جديد مثل الجدول، يقوم محرك قاعدة البيانات بإنشاء كائن تشغيل بيانات مقابل يدعى كائن TableDef لهذا الجدول بالإضافة إلى كائن تشغيل بيانات أخر يدعى كائن Document الذي يخزن معلومات إدارية عن الجدول الجديد. أيضاً عندما تعد قاعدة معلومات وخصائص تأمين، فأنت تعمل مع كائنات تشغيل البيانات. يستكشف هذا الفصل استعلام الخصائص لوصف المميزات الخاصة بالكائنات ويعلمك كيفية قراءة وإعداد وتغيير فئات مختلفة من الخصائص.

يقدم الفصل الثاني الأحداث والتي تنطلق مع الإجراءات بمجرد تفاعلك مع كائن ما فأنت تغيره. على سبيل المثال، عندما تفتح نموذجاً ما أو تحرك مؤشر الماوس لمربع نص. فإن هذا الإجراء يغير خاصية واحدة على الأقل. تكون بعض التغييرات التي يمر بها الكائن متوفرة كفرض برمجة وهذه التغييرات الخاصة هي أحداث الكائن تتكون البرمجة في أكسس من كتابة البرامج والترتيب بحيث يقوم أكسس بتشغيلها تلقائياً عندما تحدث الأحداث يقوم هذا الفصل بوصف الأحداث التي تتعرف عليها الكائنات ومعرفة الشروط المحددة والدقيقة التي تطلق الأحداث أمر مهم للغاية بالنسبة لبرمجة VBA.

تسمية الكائنات

حتى أبسط قاعدة بيانات في أكسس لها مئات العناصر مثل الجداول والحقول والفهارس والعلامات والاستعلامات والنماذج والتحكمات والتقارير والخصائص وما إلى ذلك وكل عنصر تقوم بإنشائه بنفسك أو يقوم أكسس بإنشائه لك هو كائن. والكائن مفهوم سنرجع إليه في مرات مختلفة في هذا الكتاب. وللكائن طبقات عديدة في الطبقة الخارجية. الكائن ما هو إلا شيء تستخدمه أو تغيره عن طريق إعداد الخصائص في ورقة خاصة أو حوار أو تشغيل إجراء VBA. أما بالنسبة لباقي الطبقات فسيأتي الحديث عنها لاحقاً.

يجب تنظيم مئات الكائنات في قاعدة البيانات حتى يسهل فهما يساعد إطار Database في ذلك عن طريق عرض الكائنات الأساسية والتقارير وصفحات تشغيل البيانات والماكرو والوحدات في مشروع أكسس، الكائنات المتوفرة هي الجداول وطرق العرض والإجراءات المخزنة والرسوم التوضيحية الخاصة بقاعدة المعلومات والنماذج والصفحات والماكرو

والوحدات. يمكنك تقديم مستوى أخر من التنظيم عن طريق استخدام اسم كائن لتوفير معلومات حول نوع وغاية الكائن.

ملاحظة في مشروع مايكروسوفت أكسس، يتم تخزين الجداول وطرق العرض والإجراءات المخزنة والرسوم التوضيحية الخاصة بقاعدة المعلومات في المزود اما بقية الكائنات فتم تخزينها داخل أكسس نفسه.

الأسماء التي توثق نفسها

تسمية كائن ما هي الفرصة الأولى للتعرف عليه بصورة مميزة والاسم الذي يتم اختياره من الممكن ألا يوفر أية معلومات على الإطلاق أو على العكس يمكنه أن يوثق الكائن بصورة وافية. على سبيل المثال، عندما تقوم بتسمية التحكمات في نموذج ما فلك الحرية في إطلاق أسماء رقمية مثل. Control1 أو Control2 وما إلى ذلك في المقابل، يمكنك تضمين معلومات شارحة داخل الاسم مثل txtLastName ومن هذا الاسم تستنتج بسهولة نوع التحكم والهدف منه أيضاً فلأسم يشير إلى أن التحكم هو تحكم مربع نص يحمل اسم شخص ما.

والاسم المحمل بمثل هذه المعلومة هو اسم موثق لنفسه وعندما تستخدم الأسماء التي توثق نفسها. لا تحتاج لإعداد قاموس منفصل لتعرف نوع وهدف الكائنات يمكنك صنع شفرات التسمية الخاصة بك أو تبني خطة تسمية أنشأها آخرون ومن المهم أن تستخدم نوعاً من قياس التسمية وتطبقه بصورة جيدة مع الوقت يتبع مبرمجو أكسس قياس تسمية يعتمد على نمط لتسمية الكائنات يدعى نمط Hungarian وهو على اسم يرجع لبلد مخترع هذا النمط وهو شارلز سيموني.

ملاحظة نمط قياس تسمية Hungarian تم تقديمه لمجتمع أكسس لأول مرة عن طريق ستان ليزتسينكي وجريج ريديك في مقالة لهما بعنوان (تسمية الكائنات في أكسس الإصدار الثاني من قياس مقترح) المنشورة في جريدة Smart Access في أغسطس عام ١٩٩٣ وتم تحديثها في المقالة (مراجعات لقياس تسمية ليز تسنسكي/ ريديك لاكسس ٢,٠) بقلم ليزتسيدسكي وريديك وليتوين وجبتس في جريدة Smart Access في مايو عام ١٩٩٤.

في نمط تسمية Hungarian، اسم الكائن له أربعة أجزاء: "مقطع أولي" "علامة" "اسم أساس" "مقطع نهائي".

[prefix][tag][BaseName][Suffix]

وأجزاء الاسم لها المعاني التالية:

- يعدل المقطع الأولي العلامة وغالباً ما يكون صغير واحدا للعلامة مقطع أولي واحد أو أكثر لتوفير المعلومات الإضافية عن نوع الكائن.
- تشير العلامة إلى نوع الكائن وتكون غالباً أحرف صغيرة عددها ثلاثة أو أربعة.
- الاسم الأساسي هو الاسم الذي كنت ستستخدمه على الأرجح إن لم تكن متبعاً لاصطلاح تسمية. والاسم الأساسي هو كلمة أو اثنتان تصفان هدف الكائن. اجعل الأحرف الأولى من كل كلمة مكتوبة بأحرف كبيرة ولا تستخدم مسافات. من الأفضل أن تحدد الكلمات بأكملها لكن الاختصارات ممكنة إن كانت طويلة بالقدر الكافي لتذكرها أو يسهل التعرف عليها من قبل شخص أخر يحاول فهم عملك.
- يعدل المقطع النهائي الاسم الأساسي. وهو غالباً كلمة واحدة يكون الحرف الأول منها مكتوباً كحرف كبير. من الممكن للاسم أن يكون له مقطع نهائي واحد أو اكثر لتوفير معلومات إضافية حول كيفية استخدام الكائن.

عندما تقوم بتسمية كائن، فأنت تعتمد على الاسم الأساسي والمقطع أو المقاطع النهائية أما العلامات والمقاطع الأولية فيتم تحديدهما من قوائم قياسية تقوم بإنشائها أو اقتباسها.

وما يلي علامات شائعة لكائنات إطار Database والتحكمات في النماذج والتقارير.

OBJECT	**TAG**
Table	tbl
Query	qry
Query (append)	qapp
Query (filter)	qflt
Query (union)	quni
Query (update)	qupd
Query (delete)	qdel
OBJECT	TAG
Form	frm
Form (dialog)	fdlg
Report	rpt

OBJECT	TAG
Macro	mcr
Macro (for form)	m[formname]
Macro (for report)	m[reportname]
Module	bas
Check box	chk
Combo box	cbo
Command button	cmd
Custom control	ocx
Image	img
Label	lbl
Line	lin
List box	lst
Option button	Opt
Option group	Grp
Page	Pg
Text box	Txt
Toggle button	Tgl
Subform/Subreport	Sfr
Subreport	Srp

ملاحظة

هناك نموذجان للماكرو بدون علامات وهما AutoExec وautoKeys ويجب أن يكون لهما هذه الأسماء حتى يتعرف عليهما أكسس كما تستخدم bas أيضاً كعلامة للوحدة بدلاً من mnemonic mod لحصول على ترابط مع Visual Basic.

لسوء الحظ لا يتتبع المعالجات نمط تسمية Hungarian ولا يتبع نموذج تطبيق Northwind الذي سنستخدمه عبر هذا الكتاب نمط تسمية Hungarian هو الآخر، لذلك فهذا الكتاب لا يمارس قاعدة الترابط الذي ينادي بها لقد قرر استخدام قاعدة بيانات Northwind حتى يتسنى لك العمل

مع قاعدة المعلومات نفسها المعروفة المستخدمة في توثيق أكسس بدلاً من تعلم قاعدة معلومات جديدة نتيجة لذلك، ستستخدم كلاً من الأسماء المعطاة للكائنات الموجودة وأسماء نمط Hungarian للكائنات الجديدة الموجودة وأسماء نمط Hungarian للكائنات الجديدة التي تقوم بإنشائها.

إليك أمثلة عن كيفية تغيير أسماء بعض الكائنات في قاعدة بيانات Northwind باستخدام العلامات:

OBJECT	TAGGED NAME
Customers table	TblCustomers
Customers form	FrmCustomers
Orders form	FrmOrders
Orders Subform form	FsfrOrders
Customers macro	MfrmCustomers
Customer Labels Dialog form	FdlgCustomerLabels
Customers and Suppliers by City	QuniCustomersSuppliers
CustomerID combo box	CboCustomerID
PrintInvoice command button	CmdPrintInvoice

ملاحظة انظر كتاب Access 2000 Developer's Handbook للكاتب بول ليتوين وكين جتيس ومايك جليبرت Sybex ١٩٩٩ للحصول على قائمة كاملة من العلامات الشائع استخدامها.

الأسماء في برمجة VBA

من المميزات المميزة في برمجة VBA هو قدرته على التعامل مع معلومات متغيرة. تخيل أنك تريد الاحتفاظ بحساب لعدد المرات التي تقوم فيها بتشغيل عملية ما لفعل ذلك تقوم بتعريف متغير Counter لتخزين الرقم الذي يمثل عدد المرات التي قمت بتشغيل عملية فيه ولتزويد الحساب بواحد في كل مرة تقوم بالتشغيل فيها. في برمجة VBA تقوم بتعريف أو إعلان متغير باستخدام عبارة إعلان في وحدة وعندما تعلن متغيرا ما يقوم VBA بإعداد موقعاً جانباً في لحفظ القيمة الحالية. للتعريف تقول أن المتغير هو موقع تخزين مؤقت في الذاكرة تقوم بتسميته استخدامه لحفظ قيمة أو للإشارة لكائن ما وهذه بعض المقاطع الأولية الشائعة لأنواع بيانات متغير VBA.

VARIABLE TYPE	PREFIX
Array	A
Yes/No (Boolean)	bln
Currency	cur
Date/Time	dtm
Double	dbl
Integer	int
Long	lng
Memo	mem
OLE	ole
Single	sng
Text (string)	str
Variant	var
Object	obj

يناقش الفصل الثامن متغيرات VBA ويشرح تخزين أنواع مختلفة من البيانات بتحديد أي الإجراءات يستطيع رؤية متغير (يدعى مجال المتغير) وتحديد مدة عمر المتغير في الوقت الحالي لاحظ أن المقاطع الأولية التالية استخدامها شائع لعمر ومجال متغير VBA:

- متغير مستوى إجراء محلي وعمر مستوى إجراء ليس له مقطع أولي.
- متغير مستوى إجراء محلي وعمر مستوى برنامج متغير ثابت له (المقطع الأول) S.
- متغير مستوى وحدة خاص وعمر مستوى برنامج له المقطع الأولي m.
- متغير مستوى وحدة عام وعالمي وعمر مستوى برنامج له المقطع الأولي.

تستطيع أيضاً تعريف المتغيرات للإشارة للكائنات مثل النماذج أو التقارير أو التحكمات وتدعى هذه المتغيرات المتميزة متغيرات الكائن. وفيما يلي بعض العلامات الشائعة لمتغيرات كائن VBA:

OBJECT TYPE	TAG
Application	app
Collection	col
Control	ctl

OBJECT TYPE	TAG
Controls	ctls
Form	frm
Report	rpt

في برمجة VBA، يمكنك العمل مباشرة مع Jet مستخدماً كائنات تشغيل البيانات الخاصة بـه. فيما يلي بعض العلامات الشائعة لكائنات تشغيل البيانات.

OBJECT TYPE	TAG
Database	db
Error	err
Errors	errs
Field	fld
Index	idx
Property	prp
QueryDef	qdf
Recordset	rst
Relation	rel
TableDef	tdf
Workspace	ws

ما يراه المستخدم

عندما تقوم بإنشاء تطبيق تخصيص قاعدة بيانات باستخدام التقنيات الموجودة في هذا الكتاب، فإن المستخدم يرى فقط نماذج وصفحات وتقارير يرى المستخدم تعليق النموذج أو الصفحة أو التقرير في شريط عنوان الإطار ولا يرى الاسم المشفر الذي تستخدمه لإنشاء التطبيق اجعل التطبيق معبراً عن المعلومات وقم بتضمين المسافات بشكل مناسب وتجنب استخدام اللغة المتخصصة فيما يلي بعض الأمثلة.

OBJECT NAME	CAPTION
RptMailLabelCust	Mailing Labels: Customers
CboCustomerID	Lookup Customer
FrmCustomers	Customers

تغييرات الاسم

لسوء الحظ، لا يوفر أكسس طريقة لنشر تغييرات الاسم فعندما تريد تغيير اسم جدول أو حقل أو تحكم أو نموذج أو استعلام أو ما إلى ذلك ستحتاج للمرور بكل الجداول والاستعلامات والنماذج والتقارير وصفحات تشغيل البيانات والماكرو والوحدات وعمل كل التغييرات يدوياً "لوحدات VBA، من السهل القيام بذلك باستخدام أمر Replace".

بدون أداة لنشر تغييرات الاسم يكون دليل أكسس الأساس هو "لا تقم بإجراء أية تغييرات للاسم" مع ذلك حتى مع أفضل تخطيط، من الممكن أن تحتاج لتغيير الأسماء. على سبيل المثال، عندما تستخدم Combo Box Wizard لإنشاء مربع تحرير وسرد وبحث لا تواتيك الفرصة لتسمية التحكم لأن المعالج يستخدم الاسم الفرضي Combon ويكون حرف n رقماً ولأن الاسم الفرضي لا يعرف هدف الكامل، من الأفضل تغيير الاسم لواحد يتبع مقياس التسمية ويحمل معلومات من كل من نوع الكائن والهدف منه.

ملاحظة هناك بعض الأدوات التي تساعدك في تعبيرات الاسم اثنتان من هذه الأدوات المساعدة هي Speed Ferret من إنتاج Black Moshannon Systems (هاتف ٨١٤٣٤٥٥٦٥٧) وFind و replace وهي أداة مساعدة من صنع ريك فيشر (http://www.rickworld.com/) تثبت كلتا الأداتين كإضافات وتنشران تغييرات الاسم في التطبيق الخاص بك. مع ذلك يجب الحذر عند استخدام أية أداة مساعدة لتغيير الاسم لأنها تسمح بإجراء تغييرات الاسم بسهولة شديدة وهذا يؤدي إلى عدم استطاعتك التراجع عن التغييرات التي قمت بإجرائها إذا رغبت في ذلك. يجب عليك دائماً مساندة قاعدة المعلومات قبل استخدام أداة مساعدة لتغيير الاسم.

وصف خصائص كائن

للكائنات خصائص تصف صفاتها المميزة بمعنى إن الكائن هو ناتج لجمع خصائصه المميزة في أية لحظة، كل نموذج وتحكم وحقل وعلامة وما إلى ذلك يتم تعريفهم عن طريق القيم الحالية لخصائصهم ويتم تخزين مجموعة القيم الحالية في الذاكرة كحالة الكائن. وللكائنات نوعان من الخصائص التي يمكن إعدادها:

- خصائص وقت التصميم وهي خصائص تستطيع إعدادها عندما تعمل في طريقة عرض Design وهي موجودة. في ورقة خاصية الكائن.

- خصائص وقت التشغيل وهي خصائص تستطيع إعدادها أو قراءتها فقط في وقت التشغيل، وهي غير موجودة في ورقة خاصية الكائن.

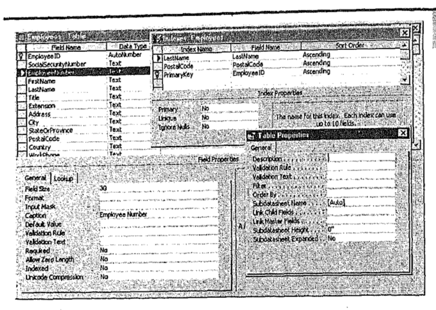
ملاحظة سنستخدم في هذا الكتاب مصطلح وقت التصميم للإشارة إلى الوقت الذي تعمل فيه في طريقة عرض Design الخاصة بكائن ما ومصطلح نوع التشغيل للإشارة إلى الوقت الذي تستعرض منه كائناً ما في طريقة عرض Datasheet أو طريقة عرض Form أو طريقة عرض Page أو طريقة عرض Print Preview ومصطلح وقت التشغيل للإشارة إلى الوقت الذي تشغل فيه إجراء VBA.

خصائص وقت التصميم

لكل كائن من كائنات إطار Database أوراق خاصية واحدة أو أكثر تسرد الخصائص التي تستطيع أنت إعدادها في وقت التصميم. على سبيل المثال، يوضح شكل ٢-١ أوراق خاصية Field وtable وIndex المتواجدة عندما تقوم بإنشاء جدول ما.

تقوم بإعداد خصائص وقت التصميم لحقل ما عن طريق تحديد الحقل في اللوح العلوي من إطار Design وإدخال قيم الخاصية في اللوح السفلي. تنقسم خصائص حقل وقت التصميم إلى فئتين مجدولتين وهما General وlookup. تتضمن خصائص General خصائص البيانات مثل حجم الحقل FieldSizeخصائص التنسيق مثل InputMask وخصائص صلاحية البيانات مثل ValidationRule وRequired. يمكن استخدام جدولة Lookup لتحديد الخصائص لحقل البحث ويجعل حقل البحث إدخال البيانات أسهل عن طريق عرض قائمة من القيم في مربع سرد وتحرير أو مربع قائمة عندما تنقر الحقل في طريقة عرض Datasheet أو تنقر تحكم نموذج مرتبطاً بحقل البحث تتضمن خصائص Lookup خصائص البيانات مثل RowSource وخصائص تصميم التحكم مثل ListRows.

بالإضافة إلى إعداد خصائص وقت التصميم باستخدام أوراق الخاصية تستطيع تغيير الإعدادات في وقت التشغيل عندما يكون إجراء VBA وهذا التشغيل. في أغلب الأوقات، يمكن تغيير إحدى هذه الإعدادات في وقت التشغيل، ويكون التغيير مؤثراً فقط خلال وقت التشغيل. على سبيل المثال، يمكن استخدام إجراء VBA لتغيير خاصية Caption الخاصة بنموذج ما، لكن عندما فتح النموذج في وقت التصميم، ستلاحظ أن القيمة المخزنة الخاصة بخاصية Caption لم تتغير.



الشكل ٢-١
أوراق خاصية Index وtable وfield في جدول طريقة عرض Design.

خصائص وقت التشغيل في برمجة VBA

بالإضافة إلى خصائص وقت التشغيل الموجودة في أوراق الخاصية الخاصة بها، معظم الكائنات لديها خصائص وقت التشغيل والتي تستطيع إعدادها أو قراءتها فقط عندما يكون إجراء VBA قيد التشغيل وخصائص وقت التشغيل غير موجودة في ورقة خاصية الكائن وفيما يلي بعض الأمثلة:

الخاصية	تطبق	وصفها
القيمة Value	التحكمات	تحدد أو تخصص النص الموجود في مربع تحكم النص أو جزء مربع النص من تحكم مربع السرد والتحرير. تحدد خاصية Value وتخصص ما إذا كان التحكم قد تم اختياره أم لا أو ما القيمة أو الجزء الذي تم اختياره من داخل التحكم.
الرسم Painting	النماذج والتقارير	تميز ما إذا كان النموذج أو التقرير قد تم إعادة رسمه أم لا (تكمل إعادة الرسم أية شاشة معلقة ويحدث ويفيد رسم الشاشة).
النص Text	تحكمات مربع النص ومربع التحرير والسرد	يعد أو يرجع النص في تحكم مربع النص أو جزء مربع النص الخاص بتحكم مربع سرد وتحرير.
إشارة مرجعية Bookmark	النماذج	يعد إشارة مرجعية تتعرف على سجل معين في مجموعة سجل النموذج المضمنة.

الخاصية	تطبق	وصفها
مختار Selected	مربعات القوائم	يقوم باختيار عنصر أو يحدد إذا كان عنصر ما تم اختياره في مربع قائمة.

تعلمت مسبقاً في هذا الفصل أن كائنات تشغيل البيانات التي تستخدمها للعمل مباشرة مع محرك قاعدة البيانات موجودة فقط في إجراءات VBA ولا توجد Design أو أوراق خاصية لهذه الكائنات وأنت تعد وتقرأ وتغير الخصائص بكتابة العبارات في إجراءات VBA.

خصائص القراءة فقط

لبعض الكائنات خصائص أقرأ فقط وهي خصائص لا تستطيع إعدادها أو تغييرها. ومن أمثلة خاصية أقرأ فقط هي خاصية نموذج Dirty تستخدم خاصية Dirty لتحديد إذا كان السجل الحالي قد تم تعديله منذ أخر مرة تم حفظه فيها. وخصائص Read-Only غير موجودة في ورقة خاصية الكائن وفيما يلي بعض الأمثلة لخصائص أقرأ فقط.

خاصية	تطبق على	وصفها
نموذج Form	النماذج Forms	تشير إلى النموذج نفسه أو إلى نموذج مرتبط بتحكم نموذج فرعي
ActiveControl التحكم النشط العد Count	الشاشة Screen النماذج المفتوحة والصفحات والتقارير	يحدد عدد النماذج المفتوحة أو عدد التحكمات في نموذج أو تقرير مفتوح يتم العد عن طريق Collection والنماذج المفتوحة والصفحات والتقارير يتم عدها لأنها موجودة في مجموعات Forms وpages وreports.
Me	النماذج والتقارير	تشير إلى النموذج أو التقرير أو إلى النموذج أو التقرير المرتبط بتقرير أو نموذج فرعي بحيث يكون إجراء VBA حالياً قيد التشغيل (فقط VBA).
CurrentObject Name	Database and Project window إطار Database وإطار object	تحدد اسم كائن إطار Database أو إطار Project النشط.

خاصية	تطبق على	وصفها
CurrentRecord	النماذج Forms	تعرف السجل الحالي الذي يتم عرضه في نموذج ما، تقابل القيمة الموجودة بين تحكمات التنقل الفرضية في النموذج
HasData	Forms and reports النماذج والتقارير	تحدد إذا كان التقرير أو النموذج مربوطان بمجموعة سجل حالية.

أوراق الخاصية الأخرى

لكائنات إطار Database/Project والكائنات التي يحتوي عليها الإطاران أوراق خاصية تستطيع فيها إعداد خصائص وقت التصميم كما تم الشرح في القسم السابق يوفر أكسس أوراق خاصية في هيئة حوارات لإعداد خصائص قاعدة المعلومات التي تعمل فيها في الوقت الحالي ولإعداد خيارات لبيئة أكسس نفسه وأيضاً لإعداد أذون التأمين. يشرح القسم الحالي الحوارات التي تستطيع استخدامها لإعداد هذه الخصائص والخيارات. وفي معظم الأحوال تستطيع إعداد الخصائص والخيارات في إجراءات VBA.

خصائص بدء التشغيل

توجد معظم خصائص بدء التشغيل في حوار Startup الذي يعرض عدد اختيارك „ Tools Startup. وكما هو موضح في الفصل الأول. عندما تعد خيارات بدء التشغيل في حوار Startup فأنت تعد خصائص قاعدة المعلومات نفسها انظر (إعداد خصائص بدء التشغيل) في الفصل الأول. بينما من الأسهل إعداد خصائص بدء التشغيل في حوار Startup يمكن إعدادها أيضاً باستخدام VBA كما سيأتي الذكر في الفصول القادمة.

ملاحظة هناك خاصية بدء تشغيل مهمة لم تعرض في حوار Startup وهي خاصية AllowByPassKey وهذه الخاصية تتيح وتبطل مفتاح Shift المستخدم لتجنب إعدادات خاصية بدء التشغيل كما هو موضح في الفصل ١٤.

خصائص قاعدة المعلومات

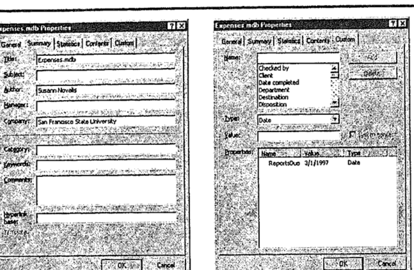
تقوم بإعداد خصائص إدارية لقاعدة المعلومات في حوار Database Properties ولعرض هذا الحوار اختر File ⇦ Database Properties أو انقر يميناً. شريط عنوان إطار Database واختر Database Properties من قائمة الاختصارات. وفي واقع الأمر، الخصائص الإدارية

لقاعدة المعلومات هي خصائص لكائنات Document عديدة يتم إدارتها بواسطة محرك قاعدة البيانات. وهناك خمس فئات مجدولة للمعلومات الإدارية وهم.

- تعرض جدولة General نفس المعلومات التي تعرض عندما تنقر يميناً اسم ملف قاعدة البيانات في Windows Explorer وتختار أمر Properties. وهذه المعلومات للقراءة فقط.
- تعرض جدولة Statistics معلومات إدارية للقراءة فقط حول تاريخ ووقت إنشاء قاعدة المعلومات الحالية ووقت وتاريخ أخر مرة ثم تعديل قاعدة البيانات وتشغيلها وطبعها أيضاً.
- تعرض جدولة Contents قائمة بأسماء كائنات إطار Database الموجودة في قاعدة البيانات وهي أيضاً للقراءة فقط.
- تعرض جدولة Summary انظر شكل ٢-٢ (أ) مربعات نص تستطيع إدخال معلومات التلخيص فيها والتي تسمح لك بالتعرف على قاعدة البيانات وتحديد موقعها بسهولة يتم تخزين المعلومة التي تدخلها كإعداد خاصية لكائن تشغيل البيانات والذي يدعى كائن SummaryInfo Document.
- توفر جدولة Custom انظر شكل ٢-٢ (ب) لك فرصة لإنشاء خصائص قاعدة بيانات مخصصة تصبح خصائص التخصيص خصائص لكائن تشغيل البيانات والذي يدعى كائن UserDefined Document.

ملاحظة يختلف إلى حد ما فتح مربع Database Properties في مشروع ما. حدد موقع قاعدة البيانات في Explorer، انقره يميناً وانقر Properties.

بالرغم من سهولة إعداد خصائص Summary وإنشاء خصائص Custom في حوار Database Properties ستتعلم كيفية إعداد الخصائص أو إنشائها مستخدماً VBA.



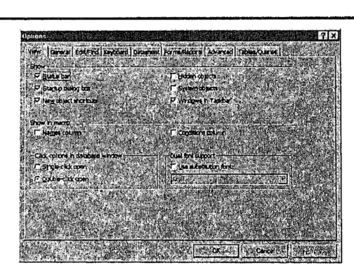
الشكل ٢-٢

"أ" جدولة Summary "ب" جدولة Custom لحوار Database Properties وهما يعرضان خصائص يمكن تغييرها.

الخيارات البيئية

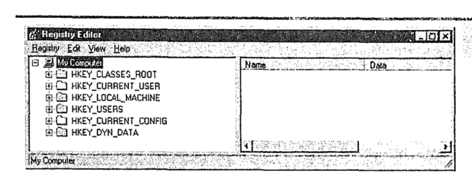
يمكنك تقليل وقت التطوير عن طريق تخصيص أكسس بيئة أكسس. وأنت تقوم بإعداد الخيارات البيئية في أكسس في حوار Options انظر شكل ٢-٣ والمعروضة عن طريق اختيار Tools ⇦ Options من قائمة Tools. وهناك ثماني فئات مجدول بها خيارات يمكن إعدادها.

تصف الخيارات البيئية صفات كائن Access Application وتبدو للوهلة الأولى كصفات لهذا الكائن. مع ذلك يتم التعامل مع الخيارات البيئية بطريقة خاصة عن طريق حفظها في قاعدة بيانات المعلومات المركزية في Windows 95 وWindows 98 وWindows NT كلاً من معلومات نظم ومعلومات بيئية وتكوينية حول التطبيقات الواحدة في ترتيب مسلسل وتدعى المجلدات الأساسية في Registry مفاتيح keys. تستطيع استخدام Registry Editor (Regedit.exe) لرؤية محتويات Registry انظر شكل ٢-٤ لتشغيل Registry Editor من سطح مكتب Windows اختر Start ⇦ Run واطبع regedit



الشكل ٢-٣

استخدم حوار Options لإعداد الخيارات البيئية.



الشكل ٢-٤
يعرض Registry Editor المجلدات التي تدعى مفاتيح والتي تحتوي على معلومات عن أجهزة الحاسب الآلي والأجهزة الطرفية وإعدادات البرامج وما إلى ذلك.

تحذير
لا تقم بأية تغييرات في Windows Registry باستخدام Registry Editor إلا إذا كنت متأكداً مما تفعل. فهذا من الممكن أن يؤدي إلى أحداث فوضى في النظام. أيضاً حتى إذا كنت متأكداً من التغييرات قم بدعم Registry قبل تحريرها بواسطة Registry Editor.

من الممكن أن تحتوي مفتاح واحد على مجلدات فرعية تدعى subkeys وعناصر بيانات تدعى value entries لرؤية محتويات لمفتاح أو المفتاح الفرعي، انقر مرتين رموز المجلد.

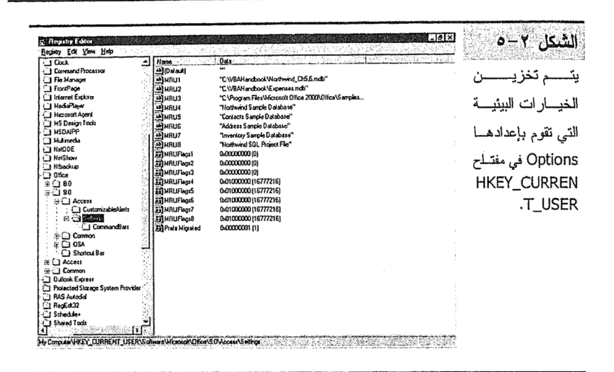
يتم تخزين إعدادات خيارات أكسس البيئية في المفتاح الفرعي بواسطة هذا المسار.

HKEY_CURRENT_USER\Software\Microsoft\Office\9.0\Access\Settings

يوضح شكل ٢-٥ إدخالات القيم للخيارات. لاحظ أيضاً إدخالات القيم للخيارات. لاحظ أيضاً المفتاح الفرعي CommandBars تحت المفتاح الفرعي subkey. يخزن المفتاح الفرعي CommandBars معلومات أمر أكسس المضمنة.

الشكل ٢-٥

يتم تخزين الخيارات البيئية التي تقوم بإعدادها Options في مفتاح HKEY_CURRENT_USER.



ملاحظة

في شكل ٢-٥ ترى إدخال Registry خاص أكسس ٨,٠ يخزن أكسس ٩٧ الخيارات البيئية الخاصة به في إدخال أكسس ٨,٠ بينما يخزن أكسس ٢٠٠٠ الخيارات البيئية الخاصة به في إدخال أكسس ٩,٠.

في إصدارات سابقة من أكسس، كانت الخيارات البيئية تخزن في قاعدة معلومات منفصلة تدعى ملف معلومات مجموعة العمل والتي يتم إنشاؤها تلقائياً عندما تثبت مايكروسوفت أكسس تستخدم الإصدارات السابقة ملف معلومات مجموعة العمل لتخزين نوعين من المعلومات وهما معلومات عن المستخدمين والمجموعات التي تقوم بإنشائها ومعلومات عن القيم التي يحددها المستخدم للخيارات البيئية. يستمر أكسس ٢٠٠٠ في استخدام ملف معلومات العمل لتخزين معلومات عن المستخدمين والمجموعات.

خصائص التأمين

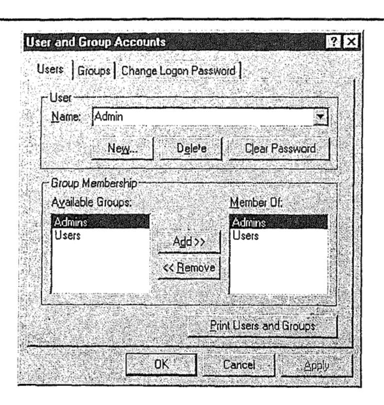
تقوم بتأمين تطبيق ما لحماية البيانات والتطبيق نفسه فأنت تحمي بنية التطبيق "وهي الجداول والاستعلامات والنماذج وصفحات تشغيل البيانات والسجلات" والبرمجة "الماكرو والوحدات" من التغييرات الناشئة عن الإهمال والتي يمكن أن تفضل التطبيق. التأمين في قاعدة بيانات أكسس هو مسئولية محرك قاعدة بيانات Jet. وJet هذا له نوعان من التأمين وهما تامين كلمة مرور قاعدة البيانات وتأمين مجموعة العمل "التأمين مختلف نوعاً ما في مشاريع أكسس حيث أن البيانات مخزنة حقيقياً في المزود".

يوجد جزءان لنوع تامين مجموعة العمل وهما المستخدمون والأذونات للمستخدمين أسماء مستخدمين وكلمات مرور تعرفهم لدى Jet كمستخدمين صالحين. لكن كائن إطار Database مجموعة من الأذونات مثل ReadData و ModifyDesign في نوع تأمين مجموعة العمل تحدد مجموعة من الأذونات لكل كائن في قاعدة المعلومات لكل مستخدم أو لكل مجموعة من المستخدمين. يتم تخزين كل من نوعي معلومات التأمين في مواقع مختلفة يتم تخزين معلومات الأذونات مع قاعدة البيانات المنفردة بينما تخزن معلومات المستخدم في ملف معلومات مجموعة العمل يثبت برنامج Setup الخاص بأكسس ملف معلومات مجموعة عمل فرضي يدعى System.mdw لكن يمكن إنشاء ملف معلومات مجموعة عمل جديد لكل مجموعة عمل. "مجموعة عمل أكسس هي مجموعة من المستخدمين الذين يشتركون في البيانات في بيئة متعددة المستخدمين".

يمكن تخزين معلومات عن المستخدمين والمجموعات وكلمات المرور الخاصة بهم في حوار User and Group Accounts انظر شكل ٢-٦. بفتح هذا الحوار اختر Tools ⇦ Security ⇦ User and Group Accounts. كائنات المستخدم والمجموعة User and Group.

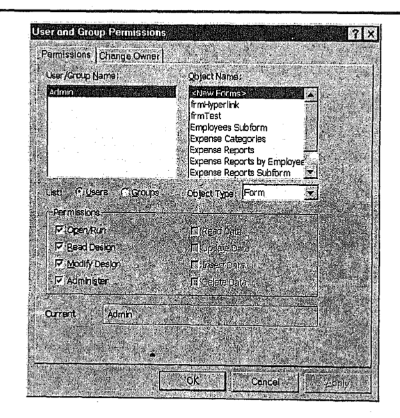
يمكنك تحديد أذونات لكل قاعدة بيانات في قاعدة البيانات الخاصة بك في حوار User and Group Permissions انظر شكل ٢-٧ لتشغيل هذا الحوار، اختر Tools ⇦ Security ⇦ User and Group Permissions.

لكل كائن إطار Database تقوم بإنشائه كائن تشغيل بيانات مقابل يدعى كائن Document يستخدم Jet كائن Document لتعقب أية أذونات قمت بإعدادها في حوار User and Group Permissions ولتعقب المعلومات الإدارية الأخرى مثل متى تم إنشاء الكائن ومن الذي قام بإنشائه. كل فئة كائن إطار Database مثل النماذج أو الماكرو لها كائن تشغيل بيانات مقابل يدعى كائن Container. يستخدم Jet كائن المحتوى Container لتعقب الأدوات التي قمت بإعدادها ولتعقب المعلومات الإدارية للفئة مثل الاسم. "كحالة استثنائية، فئتان من كائنات إطار Database وهما الجداول والاستعلامات لهما كائن Container واحد يدعى Tables". والأذونات التي قمت بإعدادها هي خصائص لكائنات تشغيل بيانات Document وكائنات تشغيل بيانات Container.



الشكل ٢-٦

ينشئ حوار User and Group Permissions المستخدمين والمجموعات وكلمات المرور الخاصة بهما.



الشكل ٢-٧

استخدم حوار User and Group Permissions لكل كائن للمستخدمين والمجموعات.

استغلال الكائنات

عندما تبدأ تشغيل قاعدة بيانات أكسس، تكون كائنات قاعدة المعلومات في انتظار ما ستفعله. وعندما تنقر زر الماوس أو تدخل ضغط مفاتيح أو تختار أمر قائمة فإن الحاسب الآلي ينفذ برامج تستغل الكائنات. وفي واقع الأمر هناك نوعان من البرامج وهما البرامج المضمنة والتي هي جزء من أكسس والبرامج التي تكتبها بنفسك. وفي كلتا الحالتين، البرامج هي تعليمات يتم تمريرها للحاسب الآلي لفعل شئ للكائنات أو لأكسس تتضمن البرامج المضمنة برامج يتم تشغيلها عندما تختار أمر قائمة مضمن أو زر شريط أدوات وهذه هي البرامج الداخلية التي تجعل أكسس يعمل. أما البرامج التي تكتبها بنفسك فهي تلك التي تشتغل عندما تختار قائمة تخصيص أو زر شريط أدوات أو عندما تنقر زر أمر تخصيص أو تفتح نموذجاً ما.

والهدف من البرامج هو استغلال الكائنات. والبرامج تفتح وتغلق النماذج وتشغل الاستعلامات لتحديد أو تعديل مجموعات من السجلات وتطبع السجلات وتحفظ التغييرات التي تمت في البيانات في ملف قاعدة البيانات البرامج تبدأ تشغيل قاعدة البيانات وتغلقها كما تبدأ تشغيل الحالات وتنهي أكسس.

طريقة أخرى لفصل البرامج عن طريق المكان الذي تخزن فيه هذه البرامج:

- تدعى البرامج المخزنة كمكونات داخلية للكائنات مطرقاً methods وتستطيع جعل كائن ما يشغل طريقة من الطرق الخاصة به وللكائنات طرق مضمنة كما تستطيع أنت أيضاً إنشاء طرق التخصيص الخاصة بك.
- تدعى البرامج المخزنة كمكونات خارجية منفصلة عن كائنات ماكرو وإجراءات macros and procedures. تطلب من أكسس أن يقوم بتشغيل ماكرو وإجراء لإحداث أعمال في الكائن.

ملاحظة تنتمي الإجراءات التي تقوم بتخزينها في وحدة نموذج أو وحدة تقرير النموذج أو التقرير الذي حددته كخاص كطرق للنموذج أو التقرير أما الإجراءات الأخرى الذي تحددها كعامة يمكن اعتبارها كمكونات خارجية "بالرغم من حفظها مع النموذج أو التقرير". انظر الفصل ٧ لمزيد من المعلومات على الإجراءات.

عندما تعمل تفاعلياً مع أكسس أو عندما تستخدم برمجة الماكرو لجعل قاعدة البيانات تلقائية، فأنت لا تعمل مباشرة مع كائنات تشغيل البيانات مع ذلك، تستطيع كتابة برامج VBA للتحكم في الكائنات التي يقوم محرك قاعدة البيانات بإدارتها.

عمر الكائنات

يعيش الكائن لفترة زمنية تدعى عمر الكائن lifetime عندما تقوم بإنشاء كائن إطار Database وتحفظ ما قمت به، يتم حفظ الكائن في ملف mdb أو adp file وهذا يعني أن كائنات إطار Database دائماً أو هي كائنات دائمة persistent تبدأ فترة عمر كائن إطار Database عندما تقوم بإنشائه لأول مرة وتنتهي عندما تحذفه من ملف mdb أو adp.

ملاحظة لقواعد بيانات أكسس نوع ملف mdb. أما المشاريع فلها نوع ملف adp.

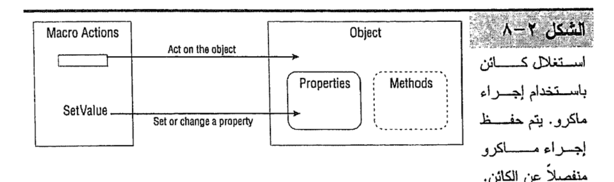
في المقابل، ليست كل الكائنات التي تتم إدارتها من قبل محرك قاعدة البيانات Jet دائمة فبعض كائنات تشغيل البيانات بما في ذلك كائنات DBEngine وerror وworkspace وrecordset هي كائنات مؤقتة أو غير دائمة ولا يتم حفظها في ملف mdb. يتم إنشاء بعض الكائنات المؤقتة تلقائياً في كل مرة يتم فتح ملف قاعدة البيانات فأنت تقوم بإنشاء كائنات مؤقتة أخرى في إجراءات VBA. وتوجد الكائنات المؤقتة في الذاكرة فقط يمكن أن تمتد الفترة الزمنية لعمر الكائن المؤقت طوال الفترة التي تمون فيها قاعدة البيانات مفتوحة أو من الممكن أن تكون فترة زمنية أقل قليلاً على سبيل المثال، تستطيع إنشاء كائن Recordset في إجراء VBA، يبقى الكائن الذي تقوم بإنشائه في أثناء تشغيل إجراء VBA الذي قام بإنشائه وعندما يتوقف أو ينتهي VBA، لا يكون لكائن Recordset وجود بعد ذلك.

استخدام الماكرو

في برمجة ماكرو، تقوم باستغلال الكائنات باستخدام تعليمات فردية تدعى macro actions. يوجد في أكسس أكثر من خمسين إجراء تم جعل إجراءات ماكرو قريبة من المثالية على حدا من أجل الأداء. تتساوى كثير من الماكرو مع أوامر القائمة والأخرى تقاد تفاعلات المستخدم اليدوية بينما توفر الماكرو غير الموجودة في واجهة تطبيق المستخدم.

تستخدم معظم إجراءات ماكرو لاستغلال كائنات إطار Database والكائنات التي تحتوي عليها مثل الحقول والأقسام والتحكمات وأنت تستخدم إجراءات ماكرو هذه لاستغلال الكائنات على سبيل المثال، تقوم إجراءات ماكرو بفتح وإغلاق وتحريك واستيراد وتصدير وحذف الكائنات. وتستخدم إحدى هذه الإجراءات وهو إجراء ماكرو SetValue لتغيير قيم صفات الكائن، كما تستخدم إجراء SetValue لتغيير قيم الحقول والتحكمات يوضح شكل ٢-٨ الطريقة التي تستخدم بها إجراءات ماكرو لاستغلال الكائنات.

لا توجد طريقة لإنشاء إجراءات تخصيص ماكرو الخاصة بك لتشغيل أكثر من إجراء ماكرو في المرة الواحدة تقوم بإنشاء ماكرو فهو مجموعة من إجراءات ماكرو كوحدة واحدة.



الشكل ٢-٨ استغلال كائن باستخدام إجراء ماكرو. يتم حفظ إجراء ماكرو منفصلاً عن الكائن.

ومن أسباب سهولة تعلم برمجة ماكرو هو أن التعليمات التي تقوم بإنشائها مستخدماً الماكرو تماثل التعليمات التي تقوم بإعطائها عندما تعمل بالتفاعل في الواقع، أفضل الطرق للبدء في كتابة الماكرو هي العمل خلال الخطوات التفاعلية لعملية ما تم ترجمة كل خطوة في إجراء ماكرو عندما تعمل تفاعلياً من الخطوات التي يجب عليك اتخاذ ذلك ما هي تحديد الكائن قبل أن تجري أي إجراء عليه. تحدد إطار مفتوحاً عن طريق النقر داخله وتحدد تحكماً عن طريق الجدولة له أو النقر داخله وتحدد سجلاً عن طريق النقر داخل محدد السجل الخاص به أو اختيار واحداً من أوامر Go To في قائمة Edit وما إلى ذلك وبرمجة ماكرو مركزية التحديد selection-centric لأن الماكرو يجب أن يحدد أي الكائنات ثم تحديدها واحدة من هذه الأشياء يجب أن تحدث قبل اتخاذ أي إجراء على الكائن.

- يمكن استخدام إجراء ماكرو مثل OpenForm وselectObject وGoToControl والتي تحدد كائناً.
- إذا كان الكائن قد تم تحديده بالفعل، يمكن استخدام خصائص الكائن مثل ActiveForm وActiveControl والتي تشير إلى خصائص الكائن النشطة.

عندما تستخدم ماكرو لاستغلال كائن ما، لا توجد طريقة مجدية لسؤال الكائن بتشغيل طريقة من طرقه. عندما تستخدم برمجة ماكرو فقط، لا تكون الطرق جزءاً من بيئة البرمجة يوضح الشكل ٢-٨ الطرق في مستطيل منقط داخل الكائن ليدل على أن الطرق موجودة لكن غير متوفرة.

يقتضي تحديد الكائن بعض الفهم لمركز الكائن. الكائن يكون مركز عندما تكون لديه القدرة على استقبال ناتج إجراء مؤشر الماوس أو لوحة مفاتيح التحكم الذي له مركز active control. وعندما يكون للتحكم مركزاً، هناك في أغلب الأوقات دلائل مرئية على سبيل المثال، زر الم

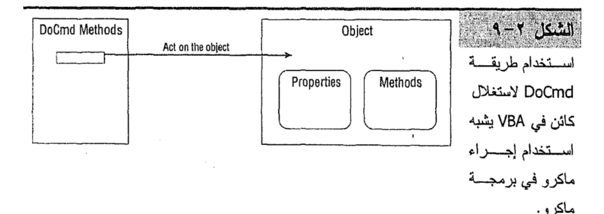
الذي له المركز موجود دخل مستطيل منقط. لا تستطيع بعض التحكمات مثل العناوين والسطور والمستطيلات والصور وفواصل الصفحة استقبال المركز خاصية Enabled وخاصية Visible يجب أن تقوم بإعداد الخاصيتين كلتيهما للسماح لتحكم بعينه باستقبال المركز.

عندما تنقر نموذجاً، يصبح نموذجاً نشطاً ويتغير لون شريط العنوان الخاص به ليدل على حالة النموذج النشطة يصبح التحكم الأول في ترتيب جدولة النموذج والذي يستطيع استقبال المركز التحكم النشط. وإذا لم يحتوي النموذج على أية تحكمات تستطيع استقبال المركز، سيستقبل النموذج نفسه المركز.

ملاحظة لمزيد من المعلومات عن الماكرو انقر Access 2000 Mastering لآلان سيميون وسيليست روبنسون Premium Edition "سايبكس ١٩٩٩".

استخدام إجراءات VBA

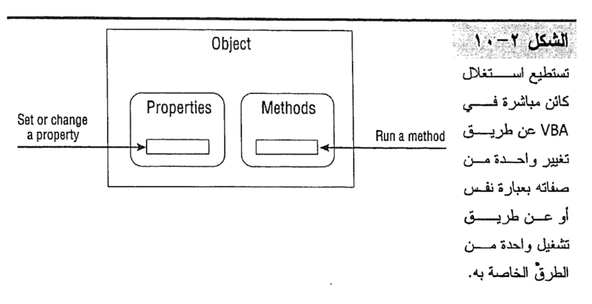
في برمجة VBA في أكسس، هناك طريقتان أساسيان لاستغلال الكائنات. والطريقة الأولى لاتخاذ إجراء على الكائن الذي تريد استغلاله هي عن طريق تشغيل برنامج خارجي عن الكائن في VBA موجود في أكسس مخصوص تدعى كائن DoCmd طرق DoCmd متشابهة مع تشغيل إجراء ماكرو في برمجة ماكرو.



الشكل ٢-٩ استخدام طريقة DoCmd لاستغلال كائن في VBA يشبه استخدام إجراء ماكرو في برمجة ماكرو.

يوفر VBA أكسس طريقة ثانية لاستغلال الكائن ويشار إليها بأسلوب إتاحة الكائن object-enabled approach وهي تتيح لك العمل مباشرة مع الكائن ومع برامجه الداخلية تستطيع استخدام عبارة تعيين لإعداد أو تغيير صفة كائن أو تستطيع جعل الكائن يشغل الطرق الخاصة به. والطرق هي مجموعة من أكثر من مائة أو إجراء محددة مسبقاً موجودة في أكسس كمكونات داخلية للكائنات في برمجة VBA، تتضمن رؤيتك لكائن كلاً من مجموعة من خصائصه التي

تعرف ما هو الكائن ومجموعة من الطرق الخاصة به والتي تعرف ما الذي يستطيع الكائن جعلـه بنفسه. يوضح شكل ٢-١٠ أسلوب إتاحة الكائن لاستغلال كائن.



الشكل ٢-١٠
تستطيع استغلال كائن مباشرة في VBA عن طريق تغيير واحدة من صفاته بعبارة نفس أو عن طريق تشغيل واحدة من الطرقُ الخاصة به.

يمكنك كتابة إجراءات VBA مركزية التحديد والتي يجب أن تحدد الكائن أو لتعرف إذا كان الكائن كائناً نشطاً يمكن إنشاء التحديد باستخدام طرق وخصائصه عديدة:

- ♦ طرق OpenForm أو SelectObject أو GoToControl وطرق كائن DoCmd أو طريقة SetFocus الخاصة بالنموذج أو التحكم الذي تريد تحديده.
- ♦ خصائص كائن مثل ActiveForm أو ActiveControl عندما يكون الكائن قد تم تحديده بالفعل.
- ♦ خاصية Me عندما تريد الإشارة للنموذج أو التقرير الذي يعمل فيه الإجراء.

يمكنك أيضاً كتابة إجراءات VBA التي تستعمل الكائن مباشرة دون تحديدها أولاً. من الصعب معرفة كيفية كتابة مثل إجراءات مركزية الكائن هذه لأنها لا تحاكي الخطوات التفاعلية المألوفة، مع ذلك متم مكافأة وقتك وجهدك بواسطة إجراءات تعمل سريعاً لأن أكسس لا يحتاج إلى وقت لتحديد الكائن. في فصول لاحقة، ستعمل مع أمثلة لتقنيات برمجة مركزية تحديد ومركزية كائن.

أحداث أكسس

حالة الكائن هي مجموعة صفاته في لحظة ما. ويتم تسجيل حالة الكائن في الذاكرة وعندما تتغير أي من الصفات تتغير الحالة أيضاً بغض التغييرات في حالة الكائن تكون فرصة لكل مقاطعة العملية التي تتبع التغيير وتدعى هذه التغييرات في الحالة أحد إنشاء. عندما نموذجاً، تغيير حالة النموذج من الإغلاق إلى الفتح ويعرف هذا التغيير في الحالة كحدث للنموذج يدعى حدث Open.

ملاحظة ليست كل التغييرات في الحالة أحداثاً. إذا غيرت لون خلفية نموذج، فأنت تغير حالة نموذج لكن هذا التغيير في الحالة لا يعرف كحدث. لا تستطيع إنشاء أحداث تخصيص.

هناك خاصية حدث مقابلة لكل حدث معرف للنموذج أو للنموذج أو لتحكم أو النموذج أو للتقرير. خصائص حدث الكائن موجودة في فئة Event في ورقة خاصية الكائن. على سبيل المثال، يوضح "شكل ٢-١١" خصائص الحدث لنموذج ما.

الشكل ٢-١١
خصائص الحدث لنموذج ما.

Form
Format | Data | Event | Other | All
On Current
Before Insert
After Insert
Before Update
After Update
On Delete
Before Del Confirm
After Del Confirm
On Open
On Load
On Resize
On Unload
On Close
On Activate
On Deactivate
On Got Focus
On Lost Focus
On Click
On Dbl Click
On Mouse Down
On Mouse Move
On Mouse Up
On Key Down
On Key Up
On Key Press
Key Preview . . . . No
On Error
On Filter
On Apply Filter
On Timer
Timer Interval 0
Has Module . . . . No

نوع برمجة أكسس

نوع برمجة أكسس هو نوع برمجة محرك بواسطة الحدث.

تقوم بإنشاء VBA الذي تريد أن يشغله أكسس عندما يتعرف الكائن على حدث وتحديد البرنامج لخاصية حدث الكائن. عندما يحدث الحدث فهو يطلق البرنامج وهذا يعني أن أكسس يقوم بتشغيل إجراء VBA المطلوب ويدعى البرنامج الذي ينفذ عندما يحدث إجراء الحدث إجراء حدث event procedure.

ملاحظة هناك نوعان من الإجراءات في VBA، وهما إجراءات الدلالة والإجراءات الفرعية وتتم الإشارة بوجه عام إلى الإجراء الفرعي المستخدم كمعالج مع الحدث كإجراء حدث بينما إجراء الدلالة المستخدم كمعالج للحدث هو إجراء دلالة حدث

عندما تعين برنامجاً لخاصية حدث، فأنت توقع بالحدث عندما تقوم بإعداد مصيدة للحدث عن طريق تعيين برنامج لخاصية لحدث أنت تقاطع المعالجة الفرضية التي يقوم بها اكسس تلقائياً متبعاً حدوث الحدث، وبعد أن يتم تنفيذ البرنامج يعود اكسس للمعالجة العادية. على سبيل المثال، تستطيع أن توقع بحدث Afterupdate الخاص بمربع بحث وتحرير وسرد عن طريق تعيين برنامج لأكسس للبحث عن السجل الذي يقابل القيمة في مربع التحرير والسرد قبل استكمال المعالجة العادية والتي هي في هذه الحالة التوقف وانتظار الإجراء التالي لبعض الأحداث لا يكتفي البرنامج بمقاطعة المعالجة الفرضية التي تتبع الحدث بل ينهي المعالجة الفرضية.

يعرف اكسس الأحداث للنماذج والتقارير وللأقسام عن النماذج والتقارير وللتحكمات في النماذج. لا توجد أحداث معرفة للجداول والاستعلامات ولا توجد أحداث معرفة أيضا لصفحات تشغيل البيانات.

ملاحظة يوجد أربعة وأربعين حدثاً معرفاً في أكسس يوضح جدولان في القرص المضغوط المرفق بهذا الكتاب "تحت table/Chapter2.pdf"طرقاً مختلفة لتنظيم الأحداث.من طرق تنظيم الأحداث هو بالنسبة لما حدث ليتسبب في الحدث. "انظر جدول "Access Events Grouped by Cause" table ولأن الحدث هو تغيير خاص في حالة الكائن، تستطيع تنظيم الأحداث تبعاً للكائن "انظر جدول Access Events Recognized by Forms, Reports, and Controls.

تتابع الأحداث

عندما يفعل المستخدم أو إجراء VBA أو الحاسب الآلي شئ ما يتم التعرف على تتابع أحداث عن طريق كائن أو اكثر. لنلق نظرة على بعض الأمثلة:

نقر زر الماوس

عندما تنقر زر الماوس الأيسر بينما يكون مؤشر الماوس على تحكم ما، يتعرف التحكم على تتابع لثلاثة أحداث:

- MouseDown عندما تضغط زر الماوس.
- MouseUp عندما تطلق زر الماوس.
- Click بعد الضغط وأطلق زر الماوس.

يتعرف التحكم على أحداث MouseDown وmouseUp بغض النظر عن أي أزرار الماوس قمت بنقرها، مع ذلك يتعرف التحكم على حدث Click فقط عندما تنقر زر الماوس الأيسر.

نقر زر أمر

عندما يستطيع التحكم استقبال المركز يتم التعرف على أحداث إضافية. على سبيل المثال، النقر البسيط على زر الماوس الأيسر عندما يكون مؤشر الماوس على زر تحكم يطلق تتابعاً لخمسة أحداث معروفة من قبل زر الأمر:

- Enter قبل أن يقوم زر الأمر فعلياً باستقبال المركز من تحكم آخر في نفس النموذج أو هو التحكم الأول في النموذج الذي يستقبل المركز عندما يفتح النموذج اولاً.
- Gotfocus بعد أن يستقبل الأمر المركز.
- MouseDown عندما تضغط زر الماوس.
- MouseUp عندما تطلق زر الماوس.
- Click عندما تضغط ثم تطلق زر الماوس الأيسر.

يتم التعرف على تتابع أحداث مماثل عن طريق تحكمات أخرى مثل مربعات النص وأزرار الخيار ومربعات القائمة ومربعات التدقيق.

تغيير النص في مربع نص أو مربع سرد وتحرير

يستطيع تغيير النص في مربع نص أو مربع النص الذي هو جز من مربع سرد وتحرير عن طريق الضغط على مفتاح ما.لا يؤدي الضغط على كل المفاتيح إلى إرسال أحرف. فعلى سبيل المثال، ضغط مفتاح Tab أو مفتاح Enter لا يؤدي إلى إرسال حرف يستخدم ويندوز مجموعة أحرف ANSI لربط المفاتيح التي على لوح المفاتيح بالأحرف المعروضة على الشاشة.

يؤدي الضغط على مفتاح مجموعة أحرف ANSI إلى إرسال حرف إلى مربع النص. إذا قم الضغط على مفتاح بتغيير النص. يتعرف مربع النص على تتابع لأربع أحداث:

- ♦ KeyDown عندما تضغط أي حرف.
- ♦ KeyPress عندما تضغط مفتاح يرسل حرف ANSI.
- ♦ Change عندما يتعرف أكسس على أن محتويات مربع نص أو جز من مربع نص في مربع تحرير وسرد قد تم تغييرها.
- ♦ KeyUp عندما تطلق مفتاحاً.

إذا لم يغير ضغط المفاتيح النص، لا يحدث حدث Change إذا غيرت النص بالفعل ثم حاولت تحديث التحكم عن طريق ضغط Enter أو عن طريق التحرك لتحكم أو سجل آخرين، ما سيحدث بعد ذلك يعتمد على ما إذا كان التحكم مربع نص أو مربع تحرير وسرد.

أحداث مربع النص

عندما يغير ضغط المفاتيح النص في مربع نص وبعد ذلك تأخذ أنت بعض الإجراءات لتحديث التحكم بالتحرك لتحكم آخر في النموذج، يتعرف مربع النص على حدثين إضافيين:

- ♦ BeforUpdate عندما يتعرف أكسس على القيمة التي تغيرت والخاصة بمربع النص تماما قبل تحديث البيانات التي تغيرت إلى مخزن السجل المؤقت.
- ♦ AfterUpdate عندما يتعرف أكسس على القيمة التي تغيرت والخاصة بمربع النص تماماً بعد تحديث البيانات التي تغيرت إلى مخزن السجل المؤقت.

أحداث مربع التحرير والسرد

عندما تغير النص في مربع النص الذي هو جزء من مربع التحرير والسرد عن طريق إرسال ضغط مفاتيح ثم تحديث التحكم بضغط Enter أو عن طريق التنقل لتحكم أو سجل آخرين، يقارن أكسس القيمة بقيم قائمة مربع التحرير والسرد. لم تكن القيمة في القائمة، يتعرف مربع التحرير والسرد على حدث NotInList ما سيحدث بعد ذلك يعتمد على إعداد خاصية LimitToList

- ♦ إذا تم تعيين خاصية LimitToList في Yes، لا يستطيع أكسس قبول التغيير ويتعرف مربع التحرير والسرد على حدث Error التتابع الكامل هو: KeyDown ⇦ KeyPress ⇦ Change ⇦ KeyUp ⇦ NotInList ⇦ Error

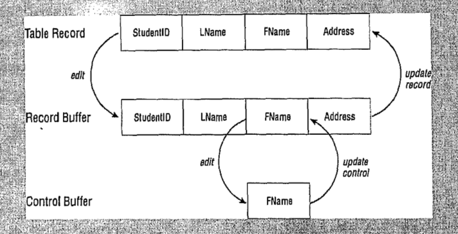
- إذا تم تعيين خاصية LimitToList في No، يقوم أكسس بتحديث تحكم مربع التحرير والسرد، ويكون التتابع المعروف من قبل مربع التحرير والسرد كما يلي: KeyDown ⇦ KeyPress ⇦ Change ⇦ KeyUp ⇦ NotInList ⇦ BeforUpdate ⇦ AfterUpdate.

يمكنك أيضا إرسال ضغط المفاتيح برمجياً باستخدام إجراء ماكروSendKey أو عبارة VBA التي تقول SendKeys يطلق إرسال ضغط المفاتيح برمجياً نفس تتابع الأحداث كما لو أنك قمت بضغط المفاتيح.

عملية التحديث

يستخدم أكسس نظام مزدوج المخازن المؤقتة لتعقب التغييرات في البيانات:

- عندما يتم عرض السجلات أولاً، يضع أكسس نسخة من البيانات الموجودة في التحكمات الخاصة به في موقع تخزين مؤقت يدعى Recordbuffer مخزن سجل مؤقت.
- عندما تطبع حرفاً ما، يصنع أكسس نسخة من البيانات التي قمت بطبعها في موقع تخزين مؤقت أخر يدعى Control buffer مخزن التحكم المؤقت



عندما تحاول الخروج من التحكم، يقارن أكسس بين البيانات الموجودة في المخزنين لتحديد ما إذا كنت قد قمت بأية تغييرات. إذا قمت بتغيير ما، يحدث التتابع التالي:

- يتعرف التحكم على حدث BeforeUpdate.

- يحدث أكسس التحكم عن طريق نسخ البيانات التي تم تغييرها من مخزن التحكم المؤقت إلى مخزن السجل المؤقت.
- يتعرف التحكم على حدث AfterUpdate. إذا لم تغير البيانات في التحكم، لا تحدث عملية تحديث التحكم.

إذا حاولت حفظ السجل، يقارن أكسس بين البيانات الموجودة في مخزن السجل المؤقت والبيانات المخزنة في الجدول. إذا قمت بتغييرات في تحكم واحد على الأقل، يحدث التتابع التالي:

- يتعرف النموذج على حدث BeforeUpdate
- يحدث أكسس السجل عن طريق نسخ البيانات التي تم تغييرها من مخزن السجل المؤقت إلى حقول الجدول
- يتعرف النموذج على حدث AfterUpdate. إذا لم تكن قد غيرت البيانات لا تحدث عملية تحديث السجل

تغيير قيمة مجموعة خيارات

عندما تغير قيمة مجموعة الخيار بنقر زر خيار أو زر تبديل أو مربع تدقيق في مجموعة الخيار، تتعرف مجموعة الخيار على التتابعات الآتية:

- BeforeUpdate عندما يتعرف أكسس على قيمة مجموعة الخيار التي تغيرت تماماً قبل تحديث البيانات التي تغيرت في مخزن السجل المؤقت.
- AfterUpdate عندما يتعرف أكسس على قيمة مجموعة الخيار التي تغيرت تماماً بعد تحديث البيانات التي تغيرت في مخزن السجل المؤقت.
- Click مباشرة بعد حدث AfterUpdate.

الجدولة من تحكم لآخر

عندما تجدول من تحكم إلى تحكم أخر في نفس النموذج دون إجراء أية تغييرات، يتعرف التحكمات على تتابع الأحداث التالي:

- يتعرف التحكم الأول على حدث Exit عندما تترك التحكم لكن قبل أن يفقد المركز.
- يتعرف التحكم الأول على حدث LostFocus بعد أن يفقد التحكم المركز.
- يتعرف التحكم الثاني على حدث Enter عندما تذهب إلى التحكم لكن قبل أن يستقبل المركز.

- يتعرف التحكم الثاني على حدث GotFocus عندما يستقبل الحدث المركز.

إذا غيرت البيانات في التحكم الأول قبل الجدولة خارجة يتعرف التحكم الأول على الحدثين التاليين قبل التتابع الموجود:

- يتعرف التحكم الأول على حدث BeforeUpdate قبل أن يتم تحديث البيانات التي تغيرت في مخزن السجل المؤقت.
- يتعرف التحكم الأول على حدث AfterUpdate بعد أن يتم تحديد البيانات التي تغيرت في مخزن السجل المؤقت.

فتح النموذج

عندما تفتح نموذجاً، يتعرف هذا النموذج على تتابع الأحداث الآتي:

- Open عندما تفتح النموذج أولاً لكن قبل أن يعرض السجل الأول.
- Load بعد أن تحمل السجلات من الذاكرة وتعرض.
- Resize عندما يتم العرض النموذج لأول مرة.
- Activate عندما يستقبل النموذج المركز ويصبح الإطار النشط "إلا أن هذا الحدث لا يتم التعرف عليه إلا إذا استقبل النموذج المركز من خلال نموذج أخر تكون خاصية PopUp الخاصة به معينة في Yes أو إذا استقبله من إطار في تطبيق أخر".
- GotFocus عندما يستلم النموذج المركز لكن فقط عندما تكون كل التحكمات في النموذج معطلة أو مختفية.
- Current قبل أن يصبح السجل الأول السجل الحالي.

عندما يكون للنموذج تحكم واحد على الأقل مرئي ومتاح، لا يتعرف النموذج على حدث GotFocus. بدلاً من ذلك يتعرف التحكم الأول المرئي على التتابع التالي مباشرة بعد تتابع حدث النموذج.

- Enter: تماماً قبل استقبال التحكم للمركز.
- GotFocus: عندما ستقبل التحكم المركز.

التوقيت هو أهم شيء

في نوع البرمجة المعتمد على الأحداث، يجب أن تقرر الحدث المناسب لإطلاق البرنامج عندما يتم التعرف على تتابع أحداث من قبل كائن أو أكثر تحتاج لتحديد حدث مناسب للبرنامج في بعض الأحيان، يكون لك الاختيار وسيؤدي تحديد أياً من أحداث عديدة إلى إتمام الإجراء كما أردت. غالباً هناك حدث واحد يعطي النتيجة التي تريدها. وتكون مهمتك كمبرمج هي تحديد هذا

الحدث الواحد الصحيح. المكان الجيد للبدء هو الحدث في التعليمات الفورية على المواضيع التالية "الأحداث، مرتبة أبجدياً" و"الأحداث، ترتيب". يوضح شكل ٢-١٢ مربع الحدث وخصائص الحدث الذين توفرها التعليمات الفورية.

ملاحظة توضح سبعة جداول على القرص المضغوط المرفق بهذا الكتاب "تحت Tables\Chapter2.pdf" الأحداث للتحكمات في النماذج "أحداث المركز وأحداث البيانات للتحكمات في النماذج" و"أحداث الماوس وأحداث لوح المفاتيح للتحكمات في النماذج" للنماذج وأقسام النماذج "أحداث الإطار والمركز للنماذج" و"أحداث البيانات والتصفية للنماذج" وأحداث الخطأ والتوقيت للنماذج وللتقارير وأقسام التقارير "الأحداث والتقارير" و"اطبع الأحداث لأقسام التقارير".

الشكل ٢-١٢ مرجع الحدث وخصائص الحدث.

Microsoft Access Help

Events and Event Properties Reference

See Also Example

The following reference groups events into categories based on the type of task and lists the events alphabetically.

Grouped by Task

Data Events	Keyboard Events
Error and Timing Events	Mouse Events
Filter Events	Print Events
Focus Events	Window Events

Listed Alphabetically

Event	Event property
Activate	OnActivate
AfterDelConfirm	AfterDelConfirm
AfterInsert	AfterInsert
AfterUpdate	AfterUpdate
ApplyFilter	OnApplyFilter
BeforeDelConfirm	BeforeDelConfirm
BeforeInsert	BeforeInsert
BeforeUpdate	BeforeUpdate

الأحداث للتحكمات في النماذج: يتعرف تحكم في نموذج على الأحداث عندما يكسب التحكم أو يخسر المركز وعندما تغير البيانات ويتم تحديث التحكم.

الأحداث للنماذج وأقسام النموذج: يتعرف النموذج على أحداث الإطار والمركز عندما تفتح أو تغلق أو تغير حجم النموذج أو عندما يكسب أو يخسر المركز. يتعرف النموذج على أحداث عندما يكسب السجل أو يخسر المركز وعندما تغير البيانات وعندما يتم

تحديث السجل وعندما تقوم بإنشاء سجل جديد أو تحذف سجل موجود وعندما تطبق أو تزيل تصفية يتعرف النموذج على حدث عندما يحدد أكسس خطأ في واجهة التطبيق أو محرك Jet وعندما يكون المركز في النموذج. أخيراً، يتعرف النموذج على حدث عندما تنقضي فترة زمنية محددة.

الأحداث للتقارير وأقسام التقارير: يتعرف التقرير على حدث عندما تفتح أو تغلق التقرير أو عندما يكسب أو يخسر التقرير المركز، أو عندما يحدد أكسس خطأ ويكون التقرير به المركز بالإضافة إلى ذلك، تتعرف أقسام التقرير على أحداث متعلقة بالتنسيق والطباعة.

إلغاء السلوك الفرضي

يعدما يتعرف الكائن على حدث ما، يقوم أكسس بالسلوك الفرضي. أحياناً، يكون السلوك الفرضي هو التوقف وانتظار إجراء المستخدم التالي. على سبيل المثال، عندما يتعرف زر أمر على حدث Click، ينظر أكسس ليرى ما إذا كنت قد كتبت شفرة VBA لخاصية حدث OnClick أم لا. إذا كنت قد كتبتها يشغل أكسس شفرة VBA وإذا لم تكن قد كتبتها يتوقف أكسس وينتظر الإجراء التالي. في أوقات أخرى، يجرى أكسس مجموعة من العمليات الفرضية قبل أو بعد تشغيل شفرة VBA المعينة للحدث. على سبيل المثال، عندما تغير البيانات في تحكم ما ثم تجدول للتحكم التالي، يتعرف التحكم الذي تغير على حدث BeforeUpdate. للتجاوب مع ذلك، يحدث السلوك الفرضي الآتي:

- يقوم أكسس بتحديث التحكم لمخزن السجل المؤقت.
- يتعرف التحكم الذي تم تغييره على حدث AfterUpdate.
- يتعرف التحكم الذي تم تغييره على حدث Exit.
- يتعرف التحكم الذي تم تغييره على حدث LostFocus
- يتعرف التحكم التالي على حدث Enter.
- يتعرف التحكم التالي على حدث GotFocus.

لبعض الأحداث بما في ذلك حدث BeforeUpdate، يقوم أكسس بتشغيل إجراء VBA المعين قبل حدوث السلوك الفرضي لهذه الأحداث، يمكنك تضمين خطوة في الأجزاء لحذف السلوك الفرضي الناتج. الأحداث التي لها سلوك فرضي يمكن حذفه موجودة في جدول ٢-١. أما بالنسبة للأحداث غير الموجودة في الجدول. يقوم أكسس بتشغيل إجراء VBA بعد القيام بالسلوك الفرضي لا يمكن إلغاء السلوك الفرضي.

الجدول ٢–١: أحداث تستطيع حذف السلوك الفرضي الناتج لها

الحدث	نتيجة السلوك الفرضي
ApplyFilter	يحذف تطبيق التصفية
BeforeDelConfirm	يحدد عرض مربع حوار تأكيد الحذف ويحذف محو السجلات. لاحظ أن الحدث AfterDelConfirm لا يزال جارياً
BeforeInsert	يحذف إدراج سجل جديد لا تستطيع إلغاء السلوك الذي يتبع حدث AfterInsert
BeforeUpdate	يحذف تحديث التحكم أو السجل لا تستطيع إلغاء السلوك الذي يتبع حدث AfterUpdate
DblClick	عندما تنقر مرتين يتبع أمر، يحدث الحدث التالي Click, DblClick, Click. يمكنك حذف Click الثانية
Delete	يحذف محو السجل
Filter	يحذف فتح إطار التصفية
Format	يحذف تنسيق القسم
NoData	يحذف طبع السجل
Open	يحذف فتح النموذج أو التقرير لا تستطيع إلغاء السلوك الذي يتبع حدث Close
Print	يحذف طبع القسم
Exit	يحذف الخروج من على التحكم لا تستطيع تحذف حدث Enter
Unload	يحذف تفريغ النموذج. لا يمكنك إلغاء السلوك الذي يتبع حدث Load

اكتساب الخبرة مع الأحداث

توضح أمثلة تتابع الأحداث كيف أنه يجب الحذر عند اختيار حدث لإطلاق البرامج. تحديد الحدث الصحيح يصبح أمراً دقيقاً عندما تعمل مع تتابع حدث متفاعل لكائنين مختلفين مثل نموذجين. لا يجب أن تكون خبيراً في الحدث حتى تبدأ مع برمجة ماكرو أو VBA، لكن كلما تعقد التطبيق، ستحتاج لدراسة نوع الحدث.

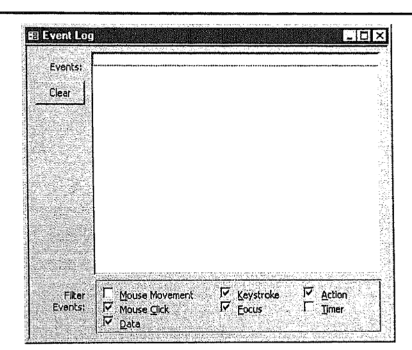
تستطيع اكتساب خبرة موجودة بسهولة ورؤية أحداث خلال حدوثها عن طريق العمـــل مـــع تطبيق Event Logger الموجود في القرص المضغوط للمرفق بالكتاب. تم إنشاء تطبيق Event Logger عن طريق مؤلفي Access 2000 Developer's Handbook (Sybex, 1999) وقد تفضل المؤلفون "بول ليتوين وكين جيتس ومايك جليـــبرت" بإعطـــاء الإذن بتضميـــن Event Logger في هذا الكتاب.

والتطبيق طريقة ممتازة لفهم تتابع الأحداث المعقدة. تستطيع استخدام التطبيق بنفسه كما ســيتم الشرح لاحقاً بالإضافة إلى ذلك، تستطيع دمج التطبيق في قاعدة البيانات الخاصة بـــك. يشــرح Access 2000 Developer's Handbook كيفية استخدام تطبيـــق Event Logger كيفيـــة استخدام تطبيق Event Logger لرؤية الأحداث في النموذج الخاص بك ويشرح إجراءات VBA التي تجعل التطبيق يعمل.

١- انسخ EventLogger.mdb لمجلد VBAHandbook في القرص الثابت الخاص بـــك. افتح قاعدة المعلومات وافتح frmLog في طريقة عرض Form انظـــر شـــكل ٢-١٣. يسجل هذا النموذج الأحداث خلال حدوثها في مربع قائمة النموذج. بعد تسجيل مجموعة من الأحداث، تستطيع مسح مربع القائمة بنقر زر Clear. يتم وضع الأحداث سوياً فـــي سبعة أنواع كما هو موضح في الأسفل. دقق أنواع الحدث التي تريد دراســتها وامســح مربعات التدقيق للأنواع الأخرى.

نوع الحدث	الحدث
Mouse Movement	MouseMove
Mouse Click	Click, DblClick, MouseDown, MouseUp
Data	AfterDelConfirm, AfterInsert, AfterUpdate, BeforeDelConfirm, BeforeInsert, BeforeUpdate, Change, Current, Delete
Keystroke	KeyDown, KeyPress, KeyUp
Focus	Activate, Deactivate, Enter, Exit, GotFocus, LostFocus

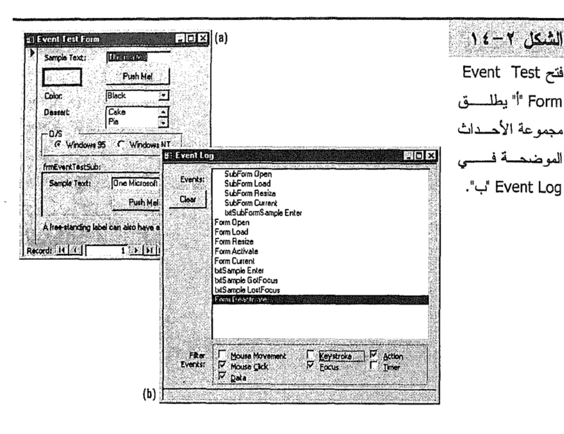
نوع الحدث	الحدث
Action	Close, Error, Load, Open, Resize
Timer	Timer



الشكل ٢-١٣
نموذج Enter Logger.

٢- دقق نزعات تدقيق Mouse Click و data و focus و Action.

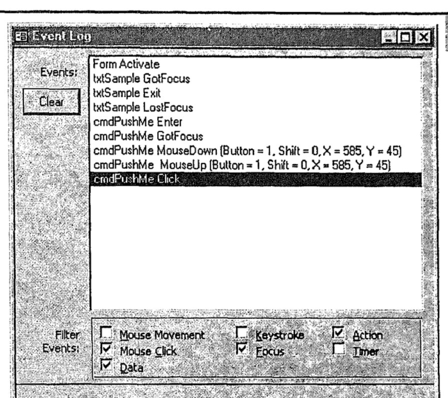
٣- افتح نموذج frmEventTest في طريقة عرض Form. يوضح شكل ٣-٢٤ Event Test Form كنا تظهر عندما تفتح النموذج لأول نرة. يحتوي النموذج على مجموعة من تحكمات الاختبار لدراسة الأحداث المرتبطة بالنموذج وبالنموذج الفرعي وبتحكمات النموذج. يشرح شكل "٣-٢٤ ب" كل الأحداث الخاصة بالأنواع التي تم تحديدها التي تحدث عندما تفتح النموذج والحدث الأول أعلى الكائنة لاحظ أن النموذج الفرعي يفتح ويحمل ا لشجة الخاصة به ويغير الحجم ويتعرف على الحدث الحالي ثم يشتبك نربع النص الموجود في النموذج الفرعي النركز. بعد حدوث أحداث النموذج الفرعي يتعرف النموذج الأشاشي على الأحداث الخاصة به في أثناء فتح النموذج ويحمل الشجة ويغير الحجم ويتعرف على أحداث Activate وgotFocus وCurrent. أخيراً يتعرف التحكم الأول في ترتيب جدول النموذج الأشاشي والذي يستطيع شتكباك النركز ونربع نص txtSample على حدث GotFocus.



الشكل ٢-١٤

فتح Event Test Form "أ" يطلق مجموعة الأحداث الموضحة في Event Log "ب".

٤- انقر زر Clear في نموذج Event Log ثم انقر زر أمر Push Me! الموجود في النموذج الأساسي. يوضح شكل ٢-١٥ أن Event Test Form يتعرف على حدث Activate ويتحرك المركز أولاً إلى مربع نص النموذج الأساسي. عندما تنقر زر الأمر، يتعرف مربع النص على حدث Exit event followed متبوعاً بحدث LostFocus. يتعرف زر الأمر على تتابع أحداث موضح في الشكل.



الشكل ٢-١٥

مسح Event Log ثم انقر زر Push Me! في النموذج الأصلي في Event Test Form يطلق أحداث يتم التعرف عليها من قبل النموذج والتحكم الأول الذي يستقبل المركز ثم زر المر.

٥- انقر زر Clear في نموذج Event Log ثم انقر زر أمر Push Me! في النموذج الفرعي يوضح شكل ٢-١٦ أن النموذج الأساسي يتعرف على حدث Activate ثم يتحرك المركز إلى التحكم الذي كان أخر من كان لديه المركز زر أمر Push Me!، يتعرف زر أمر النموذج الفرعي Push Me! على تتابع أحداث ثم يتعرف النموذج الأساسي على الأحداث الحديثة (لأن النموذج تم فتح في وضع تحرير) يتعرف تحكم النموذج الفرعي على حدث Enter، يتعرف مربع النص في النموذج الفرعي على حدث Exit ثم يتعرف زر أمر Push Me! في النموذج الفرعي على تتابع الأحداث الخاص به.

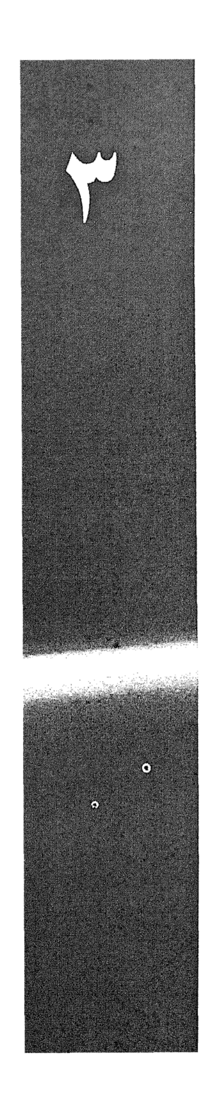
٦- استكشف الأحداث التي تم التعرف عليها من قبل تحكمات أخرى في Event Test Form. على سبيل المثال، انقر زر خيار Windows NT لرؤية الأحداث التي تم التعرف عليها بواسطة أزرار الخيار ومجموعة الخيار، انقر السهم الذي يشير لأسفل لمربع التحرير والسرد واختر عنصراً أخر، حرر النص في مربعات النص في النموذج الأساسي والنموذج الفرعي وما إلى ذلك.

خلاصة

قدم هذا الفصل مفهوم الكائنات فيما يلي النقاط الهامة:

- اختيار مصطلح اسم مثل نمط Hungarian يجعل الكائنات موثقة لنفسها عن طريق توفير معلومات عن نوع الكائن والهدف منه
- للكائنات خصائص تصف صفاتها المميزة لمعظم كائنات أكسس خصائص وقت التصميم تستطيع إعدادها في أوراق الخاصية ولكائنات أكسس وMSDE وJet خصائص وقت تشغيل تستطيع إعدادها أو قراءتها فقط عندما يكون الإجراء قيد التشغيل. بعض خصائص وقت التشغيل للقراءة فقط.
- بينما تكون بعض الخصائص متوفرة في كل من برمجة ماكرو وVBA، هناك الكثير من الخصائص التي تستطيع استخدامها في برمجة VBA فقط.
- لقاعدة البيانات نفسها خصائص تستطيع إعدادها مستخدماً مربعات حوار موجودة عن طريق اختيار أوامر قائمة تستطيع إعداد بدء التشغيل ومعلومات قاعدة البيانات العامة وخصائص التامين.
- يمكنك إعداد الخيارات البيئية. يتم حفظ الإعدادات في ملف قاعدة البيانات معلومات مجموعة عمل منفصل.
- من سمات نوع برمجة VBA المهمة هبي أنه يشغل طرق الكائن. والطريقة هي برنامج مضمن محفوظ داخلياً مع الكائن والذي تستطيع تشغيله لاتخاذ إجراء ما على الكائن.
- تعرف الإعدادات لكل خصائص الكائن حالته في لحظة ما يؤدي المستخدم أو الإجراء أو إجراء الحاسب الآلي إلى تغيير في الحالة تدعى بعض التغييرات في الحالة أحداثاً وهي متوفرة كفرص برمجة وحفظ تحكمات النموذج والنموذج وأقسام النموذج والتقارير وأقسام التقرير يتعرفون على الأحداث فعلياً، يطلق إجراء مستخدم واحد تتابع أحداث واختيار حدث مناسب من الإجراء مهارة مهمة.

والآن بعد معرفة مفهوم الكائن، حان وقت معرفة مميزات نوع كائن أكسس. يركز الفصل التالي على الكائنات والخصائص الموجودة في برمجة VBA.



تقديم نموذج وحدة لأكسس

عندما تجعل عمليات قاعدة بيانات عملية آلية، فأنت بذلك تقوم بإنشاء إرشادات تعمل عندما يتعرف كائن على حدث. يجب معرفة أي الكائنات يمكن كتابة إرشادات لها وأي الأحداث التي يتعرف عليها الكائن وكذلك كيفية كتابة الإرشادات. في الفصل السابق تم تقديم مفهوم الكائن كشيء يمكن استخدامه أو تغييره بواسطة إجراء VBA أما هذا الفصل يركز على الكائنات المُحددة أي الكائنات المتاحة وعلى الخصائص التي يمكن تغييرها وكيفية ربط الكائنات بعضها ببعض وكيفية تعريف كائن عند كتابة برنامج.

يعتبر نموذج كائن أكسس كبير ومُعقد. لذلك يهدف هذا الفصل لتهيئتك للبدء بالنموذج وذلك بتقديم هذه الكائنات والخصائص المتاحة في برمجة VBA.

ربط الكائنات ببعضها

ليس من الجديد التحدث عن كائنات نافذة قاعدة البيانات "Database" لتطبيقات أكسس المتفاعلة وهي الجداول والاستعلامات والنماذج وصفحات الوصول إلى البيانات والتقارير. سنستمر في استخدام كلمة كائن وسنضيف مزيد من الكائنات إلى قائمتنا وهي حقو لاجداول وحقول الاستعلامات وعناصر تحكم النماذج وعناصر تحكم التقارير. عند العمل مع تطبيقات أكسس بشكل تبادلي، فليس من الضروري الاهتمام بكيفية ربط الكائنات ببعضها إذ يعتبر من مهام واجهة مستخدم أكسس الاعتناء بهذه العلاقات. ومع ذلك، لإنشاء برامج تستخدم الكائنات يجب عليك معرفة كيفية ربط الكائنات ببعضها حتى يمكنك استخدام هذه العلاقات للرجوع إلى كائن في برنامج.

فهم مجموعات الكائنات سواء كانت تلك الكائنات أشخاص أو أقسام سماوية أو كائنات قاعدة بيانات. يعني صممنا فهم كيفية ربطها ببعض معاً في علاقات. وهناك نوعان من هذه العلاقة: بعض الكائنات مشابهة لكائنات أخرى وبعضها يتضمن كائنات أخرى.

الكائنات المتشابهة

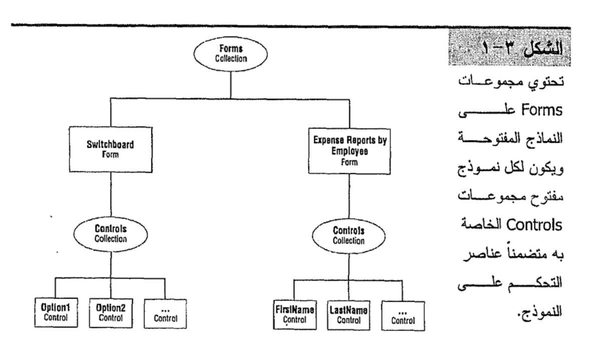
من الطبيعي جمع الكائنات ذات الخصائص والسلوك المتشابه. على سبيل المثال، من الطبيعي تجميع النماذج في قاعدة بيانات أو أزرار الأوامر أو مربعات النص. تسمى مجموعة الكائنات المتشابهة باسم مجموعة.

مجموعات الكائنات

في تطبيقات أكسس تكون أغلب الكائنات موجودة في مجموعات. على سبيل المثال، تحتوي قاعدة بيانات على مجموعة جداول واحدة تتضمن كل الجداول في قاعدة البيانات ويكون لكل جدول

مجموعة حقول تحتوي على كل الحقول المحددة للجدول، تحتوي قاعدة بيانات على مجموعة نماذج مفتوحة واحدة ويكون لكل نموذج مفتوح مجموعة عناصر تحكم تتضمن كل عناصر التحكم تم وضعها على النموذج. ويبدأ أكسس اسم كل نوع من الكائنات بحرف كبير مثل كائنات Form و Control و field. وكذلك يقوم أكسس بتسمية مجموعة بإضافة الحرف S إلى اسم نوع الكائن في المجموعة، على سبيل المثال، تحتوي مجموعة Controls لنموذج مُحدد على مجموعة Control الموضوعة على النموذج. يعتبر أكسس المجموعة نفسها كائناً على سبيل المثال. تعتبر مجموعة Controls كائناً يتضمن كائنات Control لنموذج أو تقرير مُحدد.

مثال آخر على المجموعات يرتبط بالاختلاف من نموذج مفتوح وآخر مغلق. يعتبر النموذج المفتوح كائن Form وبالتالي تكون مجموعة النماذج المفتوحة في مجموعة Forms. وبالعكس، لا يعتبر النموذج المغلق كائن Form وبالتالي ليس عضواً من مجموعة Forms وإنما يكون مجرد نموذجاً مغلقاً. يصور الشكل ٣-١ مجموعة Forms لتطبيقات Expenses عندما تكون Switchboard وExpense Reports بواسطة نماذج Employee هي النماذج الوحيدة المفتوحة.



الشكل ٣-١ تحتوي مجموعات Forms على النماذج المفتوحة ويكون لكل نموذج مفتوح مجموعات Controls الخاصة به متضمناً عناصر التحكم على النموذج.

تعتبر الكائنات المفردة هي كائنات ليست ضمن في مجموعة. على سبيل المثال، يمثل كائن Application تطبيق أكسس ويمثل DBEngine محرك قاعدة بيانات جيد. ويعتبر كل منهما كائناً مفرداً لأن لأكسس كائن Application واحد فقط ولمحرك قاعدة بيانات Jet كائن DBEngine واحد فقط. يعتبر كذلك كائن مفرداً. على سبيل المثال، يوجد كائن مجموعة Forms واحد فقط في التطبيق ولكل كائن Form في المجموعة توجد مجموعة Controls واحدة فقط. ويعتبر الكائن مُهماً عن الإشارة إليه سواء كان مفرداً أو موجوداً في مجموعة.

الفئات والأمثلة

هناك طريقة أخرى للنظر إلى مجموعات الكائنات المتشابهة وهي التفرقة بين تعريف المجموعة وتعريف الكائنات في المجموعة. يتم استخدام كلمة "فئة" للإشارة إلى تعريف مجموعة. على سبيل المثال، فئة النماذج أو فئة مربعات النص. تعتبر الفئة مخطط الكائنات. وكمثال بسيط، يمكن اعتبار أسلوب عرض جدول Design كممثل لفئة الجداول عند إنشاء جدول مُعيّن بتحديد حقول الجدول وإعداد خصائصه يمكنك إنشاء مثال للفئة.

وكمثال آخر، تمثل أداة Text Box في مربع الأدوات فئة مربع الأدوات عند استخدام أداة Text Box لإنشاء مربع نص بمجموعة خصائص مُعيّنة يمكنك بذلك إنشاء مثال.

ولهذا، عند النظر إلى مجموعة الكائنات ذات الخصائص والسلوكيات المتشابهة نجد هناك قسمين هما: أولاً تعريف المجموعة وهي الفئة وثانياً الكائنات نفسها وهي الأمثلة.

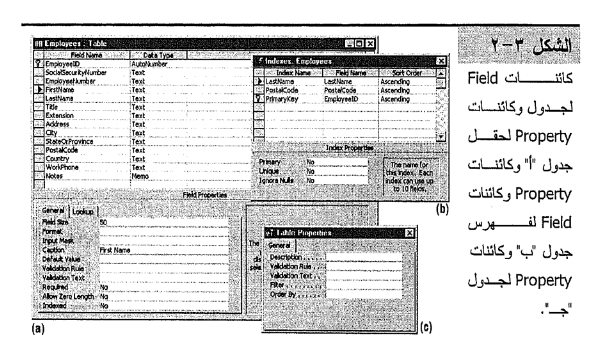
كائنات متضمنة لكائنات أخرى

تعتبر علاقة الكائنات المحتوية على كائنات أخرى هي ثاني أهم علاقة بين الكائنات. على سبيل المثال، يحتوي النموذج على عناصر التحكم الخاصة به ويحتوي الجدول على حقوله. يحتوي الجدول كذلك على فهارسه ولكل فهرس حقوله الخاصة، بصفة عامة تحتوي الكائنات على كائنات تحتوي بدورها على كائنات أخرى وهكذا. تعتبر علاقة الاحتواء هذه علاقة الرئيسي والفرعي بحيث يكون الكائن رئيسي للكائنات التي يحتويها والتي تكون فرعية للكائن المتضمنها. على سبيل المثال، يكون النموذج رئيسي لعناصر التحكم التي يحتوي عليها بينما تكون عناصر التحكم هذه هي الفرعي للنموذج.

يمكن عرض المستويات المختلفة لعلاقات الاحتواء لصفوف في سلسلة. على سبيل المثال، سلسلة علاقات الاحتواء للجداول. في أكسس يسمى كائن الجدول الذي تم تعريفه في أسلوب عرض الجدول Design باسم كائن TableDef والذي يعتبر أحد كائنات الوصول إلى البيانات التي يتم إدارتها بواسطة Jet. ويحتوي الجدول على ثلاثة مجموعات:

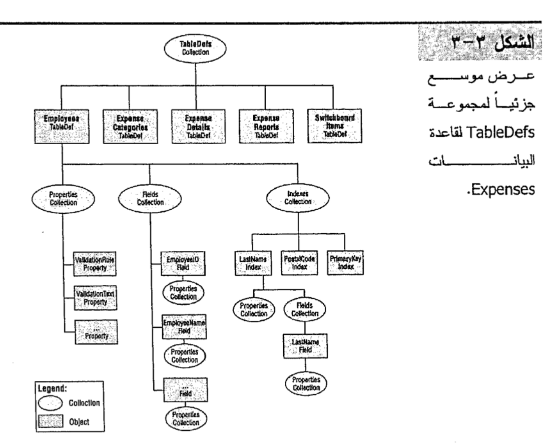
- مجموعة Field المحتوية على كائنات Field الخاصة بها. كما هو مسرود في اللوح الأعلى لأسلوب عرض Design "انظر الشكل ٣-٢ أ" سيكون لكل كائن Field مجموعة Properties المحتوية على كائن Property لكل خاصية تم سردها في اللوح الأسفل لأسلوب عرض Design وكذلك الخصائص الأخرى المتاحة في التعليمات البرمجية.

- مجموعة Indexes كما هو مسرود في حوار Indexes "انظر الشكل ٣-٢ ب" يشتمل كل كائن Index في المجموعة على مجموعة Properties المحتوية على كائن Property لكل خاصية تم سردها في اللوح الأسفل من حوار Index ويكون لكل كائن Index مجموعة Fields المتضمنة كائنات Field المسرودة في اللوح الأعلى لحوار Index. "في هذا المثال، يحتوي كل كائن Index على حقل واحد". ولكل كائن Field في Index مجموعة Properties مشتملة على كائنات Property لكائن Field.
- مجموعة Properties المحتوية على كائنات Property كما هو مسرود في حوار Table Properties "انظر الشكل ٣-٢ ب".



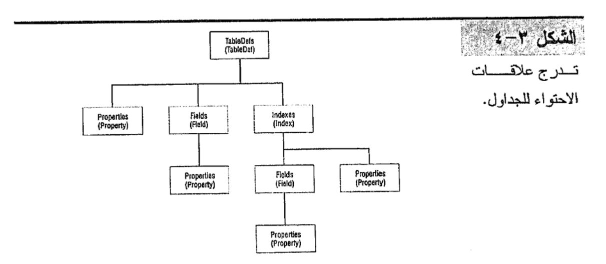
الشكل ٣-٢ كائنات Field لجدول وكائنات Property لحقل جدول "أ" وكائنات Property وField فهرس جدول "ب" وكائنات Property لجدول "جـ".

يوضح الشكل ٣-٣ عرض موسع جزئياً لمجموعة TableDefs الخاصة بقاعدة بيانات Expenses. في هذا الشكل، تم توسيع جدول Employee لعرض مجموعاته الثلاثة والتي تم توسيعها لعرض بعض أعضائها. وفي كل حالة يتم توسيع أعضاء مجموعة لعرض مجموعاتها والتي بدورها تم توسيعها لعرض أعضائها وهكذا حتى يصبح سريعاً عرض علاقة الاحتواء الموسع كبيراً وممتداً وبالتالي يكون المفهوم المهم هنا هو بنية التدرج.



الشكل ٣-٣ عرض موسع جزئياً لمجموعة TableDefs لقاعدة البيانات Expenses.

يوضح الشكل ٣-٤ عرض مطوي تماماً لتدرج الجدول الذي يركز على الهيكل. في هذا الشكل، يعرض كل مستطيل مجموعة ويتضمن عضو ممثل للمجموعة الموضعة بين الأقواس. تذكر أنه يمكن توسيع كل مجموعة لعرض أعضائها المُحددة.



الشكل ٣-٤ تدرج علاقات الاحتواء للجداول.

يجب عليك معرفة علاقات الاحتواء لكل الكائنات في أكسس لأنه عند كتابة برنامج للتحكم في خصائص سلوكيات الكائن يجب الإشارة إلى كل الكائنات التي تقع بطول طريق التدرج إلى الكائن.

تقديم أسلوب البناء لأكسس

عند تثبيت أكسس، يتم بذلك تثبيت مكونان رئيسيان هما Access Application Layer ومحرك قاعدة بيانات Jet. وللعمل باستخدام مشاريع اكسس على Microsoft SQL Server يجب عليك تثبيت "Microsoft Database Engine "MSDE.

طبقة التطبيق Application

تتكون طبقة Application من كل الملفات الضرورية للتحكم في واجهة المستخدم وكل الملفات المطلوبة لكتابة وتشغيل إجراءات VBA. وتحتوي طبقة Application على أشرطة القائمة. أشرطة الأدوات والنوافذ لإنشاء وعرض كائنات نافذة قاعدة البيانات.

عند إنشاء قاعدة بيانات بشكل تفاعلي. يمكنك العمل مباشرةً في طبقة Application باستخدام نافذة Design لإنشاء الجداول الفردية والاستعلامات والنماذج وصفحات الوصول إلى البيانات والتقارير وكذلك الماكرو والنماذج التي تدمج الكائنات في تطبيق واحد. وبالرغم من استخدام واجهة طبقة Application لإنشاء كل كائنات نافذة Database السبعة. يتم تعريف صفحات الوصول للبيانات والنماذج والتقارير والوحدات فقط ككائنات Application. وتعتبر الجداول والاستعلامات التي تم إنشائها في واجهة اكسس هي كائنات Jet للوصول إلى البيانات.

عند إنشاء مشروع اكسس فبالضرورة تقوم بإنشاء تطبيق عميل/خادم والذي يعمل مع قاعدة بيانات مثل Microsoft SQL Server. وبدلاً من استخدام محرك قاعدة البيانات Jet. يقوم المشروع باستخدام MSD الجديد. توفر المشاريع طور وصول محلي إلى قاعدة بيانات خادم من خلال أسلوب البناء OLEDB Component. يمكن اعتبار OLEDB كنوع من الوسطاء بين مشروع Application والخادم. وتقوم المشاريع باستخدام ٩ مشاريع وهي الجداول وأساليب العرض والإجراءات المُخزنة والرسوم البيانية لقاعدة البيانات والنماذج والتقارير والصفحات وصفحات الوصول للبيانات والماكرو والوحدات. ويمكن الفارق الكبير بين مشروع وقاعدة بيانات في موقع البيانات الفعلية المُخزنة في الجداول، في المشروع يتم تخزين الجداول على خادم. بينما في قاعدة البيانات يتم تخزين الجداول محلياً في أكسس نفسه. بمجرد الاتصال بقاعدة بيانات الخادم، يمكنك عرض وإنشاء وتعديل وحذف بيانات وبذلك يكون العمل مع مشاريع اكسس مشابهة للعمل مع قاعدة بيانات اكسس.

ملاحظة يجب تثبيت MSDE باستخدام ملف إعداد مختلف، قم بإضافة القرص المدمج لتثبيت اكسس وانقر نقراً مزدوجاً فوق SETUPSQL.EXE في مُجلد \Sql\x86\Setup لتثبيت MSDE.

محرك قاعدة البيانات Jet

عند العمل مع قاعدة بيانات اكسس يتألف محرك قاعدة البيانات Jet من ملفات ضرورية لإدارة البيانات وللتحكم في الوصول إلى البيانات في ملف قاعدة البيانات ولتخزين الكائنات التي تنتهي لطبقة Application. يتضمن Jet البرامج الداخلية لستة وظائف أساسية لإدارة قاعدة البيانات وهي:

تعريف البيانات وتكاملها: يمكنك باستخدام Jet إنشاء وتعديل الكائنات التي تحتفظ بالبيانات. يمكنك استخدام كلاً من الواجهة وبرمجة VBA لإنشاء وتعديل قاعدات البيانات والجداول والحقول والفهارس والعلاقات والاستعلامات. يقوم Jet بتقوية قوانين الكيان والتكامل المرجعي التي قمت بتحديدها عند تصميم الجداول واستثناء العلاقات.

تخزين البيانات: يستخدم Jet طريقة تسمى Indexed Sequential Access Method "ISAM" لتخزين البيانات في نظام الملف. وباستخدام ISAM يتم تخزين البيانات في صفحات بحجم ٢ كيلو بايب محتوية على سجل أو أكثر وسجلات بأطوال متعددة وسجلات يمكن ترتيبها باستخدام فهرس.

استدعاء البيانات: يقدم Jet طريقتين لاستدعاء البيانات. أحدهما استخدام محرك الاستعلام القوي لـ JET الذي يستخدم SQL لاستدعاء البيانات. والطريقة الأخرى هي للوصول إلى البيانات بطريقة مبرمجة باستخدام كائنات الوصول للبيانات في إجراءات VBA.

التحكم في البيانات: يمكنك باستخدام Jet إضافة بيانات جديدة وتعديل أو حذف البيانات الموجودة. يمكنك التحكم في البيانات إما باستخدام محرك استعلام Jet عن طريق الاستعلامات الإجرائية لـ SQL أو باستخدام كائنات الوصول إلى البيانات في إجراءات VBA.

التأمين: يحتوي Jet على نموذجين تأمين متضمناً نموذج كلمة سر قاعدة البيانات لتأمين بسيط لكلمة سر قاعدة البيانات بأكملها، ونموذج تأمين مجموعة عمل الذي يكون فيه تصريح للمستخدمين والمجموعات لمشاريع قاعدة البيانات الفردية.

مشاركة البيانات: يقوم Jet بتمكين عدة مستخدمين من الوصول إلى البيانات وتعديلها في نفس قاعدة البيانات. ويقوم Jet بإغلاق البيانات في صفحة معطاة عندما يقوم المستخدم بتعديل السجل. يقوم Jet بإغلاق الصفحة بمجرد بدء مستخدم بالتحرير "إغلاق تشاؤمي". بينما يقوم بفتح الصفحة بعد الانتهاء من التحرير أو يسمح Jet لعدة مستخدمين لتحرير سجل ثم يقوم بإغلاق الصفحة فقط عندما يحاول مستخدم حفظ

التغييرات أو تأكيدها "إغلاق تفاؤلي". ربما يؤدي إغلاق صفحة كاملة إلى إغلاق سجلات متعددة وذلك لأن الصفحة تحتوي على ٢ كيلو بايت من البيانات. يتضمن أكسس ٢٠٠٠ طور جديد مفرد لغلق السجل متجنباً هذه المشكلة، يمكنك اختبار هذا الطور من تبويب Advanced لمربع الحوار Tools ⇦ Options.

تحذير لا تخلط بين صفحات الوصول إلى البيانات وصفحات الجدول. صفحات الوصول إلى البيانات . صفحات الوصول إلى البيانات هي جزء من نموذج الكائن ويمكن استخدامها لعرض بيانات التقرير باستخدام الإنترنت أو الإنترانت، بينما صفحات جدول هي ٤ كيلو بايت مقطع للبيانات ويتم استخدامها بواسطة الجداول للتخزين. وبيساطة يتم الإشارة إلى صفحات الوصول إلى البيانات بكلمة صفحات.

محرك قاعدة بيانات مايكروسوفت "MSDE"

عند العمل مع مشروع أكسس. سيسمح MSDE لطبقة Application بالاتصال بعدة خوادم وإدارة البيانات. ثم تسمية المشروع هكذا لأنه لا يحتوي على أي بيانات أو كائنات تعريف بيانات. ولا يحتوي على أو أساليب عرض أو رسوم بيانية لقاعدة البيانات أو إجراءات مُخزنة. يتم تخزين قاعدات البيانات هذه في قاعدة بيانات الخادم ولكنها تتأثر بكائنات أخرى لمشروع مثل النماذج والتقارير وصفحات الوصول إلى البيانات والماكرو والوحدات.

وتكمن ميزة MSDE في تقديمه لمخزن بيانات محلي يتفق مع خادم المضيف أو يمكن أن يكون بمثابة حل لتخزين البيانات عن بُعد. يقوم MSDE بنفس الوظائف "وهي تخزين البيانات واستدعائها والتحكم فيها والتأمين ومشاركة البيانات" وذلك باستخدام وظائف خادم الضيف مثل Microsoft SQL Server Design Tools ثم دمج Design Tools بطريقة ملائمة في مايكروسوفت أكسس، ويكون جاهزاً متى قمت بإنشاء جدول جديد أو أسلوب عرض أو رسم بياني لقاعدة بيانات أو إجراءات مُخزنة في مشروع أكسس.

تدرجات الكائن

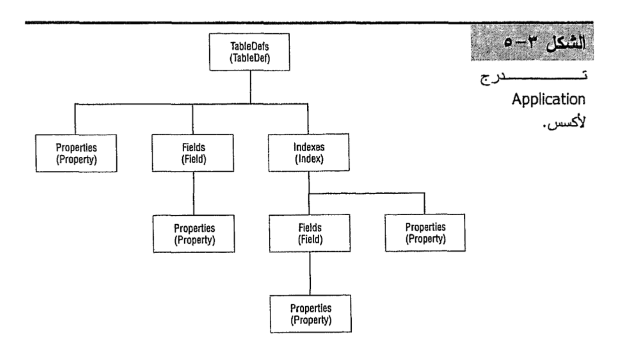
يتم إنشاء إجراءات VBA لاستخدام الكائنات، وتكون الكائنات المتاحة للاستخدام هي كائنات متضمنة. قام مطورو أكسس بتعريفها. يتم تجميع هذه الكائنات المُضمنة في مجموعاتها المتاحة وتنظيمها في تدرجات منفصلة. في كل حالة يحتل كائن مفرد أعلى التدرج يكون كائن

Application في رأس تدرج Application لأكسس وكذلك يأتي كائن DBEngine على قمة تدرج كائن قاعدة البيانات لـ Jet وكذلك كائن Current Project وCurrent Data في مقدمة تدرج محرك قاعدة البيانات MSDE ويكون كلاً من تدرجات محركات قاعدة البيانات كائنات لتدرج كائن Application.

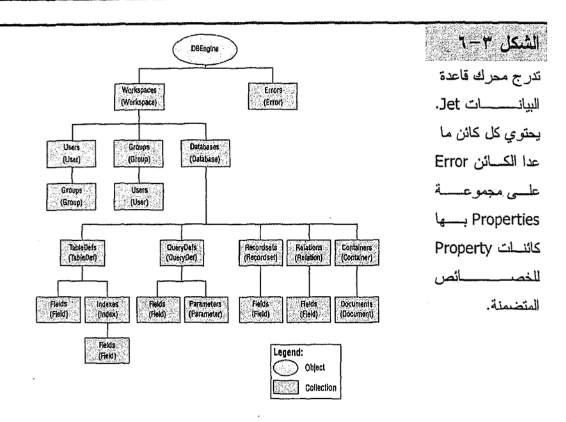
يوضح الجزء الأعلى من الشكل ٣-٥ تدرج Application المستخدم للبرمجة في VBA.

الشكل ٣-٥

تدرج Application لأكسس.

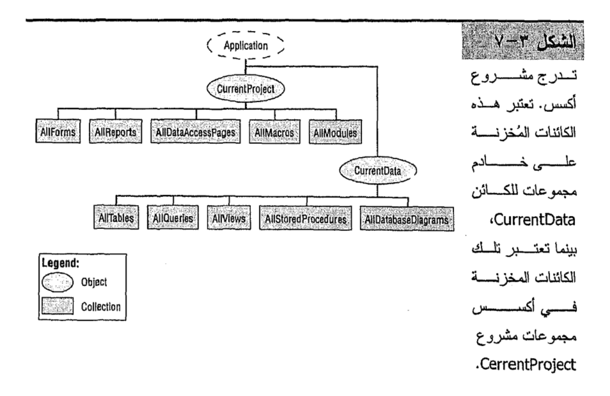


يعرض الشكل ٣-٦ كائنات الوصول إلى البيانات في تدرج محرك قاعدة البيانات بجانب المستخدم للتحكم في محرك Jet باستخدام برمجة VBA عن طريق Data Access Objects "DAO" كائنات الوصول إلى البيانات. وتعتبر DAO أكثر عناصر التحكم تفصيلاً في محرك Jet ولكن تم استبدالها بواسطة "ADO" ActiveX Data Objects. يسمح لك أكسس ٢٠٠٠ ببرمجة محرك Jet باستخدام DAO أو ADO. يتم استخدام ADO فقط لبرمجة MSDE. سوف تعرف المزيد عن DAO وADO في الفصل ٦.



الشكل ٣-٦
تدرج محرك قاعدة البيانات Jet. يحتوي كل كائن ما عدا الكائن Error على مجموعة Properties بها كائنات Property للخصائص المتضمنة.

يوضح الشكل ٣-٧ كائنات الوصول إلى البيانات في تدرج الكائن لأكسس. في هذه الأشكال يتم عرض الكائنات المفردة والتي ليست كائنات مجموعة في شكل بيضاوي بينما تظهر كائنات المجموعة من مستقبلات.



الشكل ٣-٧
تدرج مشروع أكسس. تعتبر هذه الكائنات المُخزنة على خادم مجموعات للكائن CurrentData، بينما تعتبر تلك الكائنات المخزنة في أكسس مجموعات مشروع CerrentProject.

فهم نموذج كائن التطبيق لأكسس

يتضمن نموذج كائن التطبيق Application لأكسس عدة كائنات موجودة في برمجة VBA فقـط. تتضمن هذه الكائنات كلاً مــن الكائنــات DoCmd وmodule ومجموعــة Modules وكــائن Reference. ستتعرف المزيد عن هذه الكائنات في الفصل ٥.

كائن التطبيق

يتمثل كائن التطبيق Application تطبيق أكسس نفسه وكذلك البيئة التي يتم فيها تشغيل إجراءات VBA "والماكرو". وتؤثر خصائص كائن Application في بيئة أكسس بأكملها. يقوم الجدول ٣-١ بسرد خصائص كائن Application.

الجدول ٣-١: خصائص كائن Application

الخاصية	نوع البيانات/الوصــول	البيان
Current Object Name	قراءة فقط/سلسلة	يقوم أكسس بإعدادها لسلسلة تعبير محتوية على اسم الكائن النشط ويتــم اســتخدامها لتحديد اسم كائن نــافذة قــاعدة البيانـــات النشط.
Current Object Type	قراءة فقط/عدد صحيح	يقوم أكسس بإعداد هذه لأحــد الثوابــت الأساسية في VBA. يتم استخدامها لتحديــد نوع كائن نافذة قاعدة البيانات النشط.
Hwnd Access App	قراءة فقط/طويل	يقوم أكسس بإعدادهــا. يتــم اســتخدامها لتحديد العدد الصحيــح الطويــل الفريــد "يسمى مرجع النــافذة" الــذي خصــص ويندوز لنافذة أكسس الرئيسية.
Menu Bar	قراءة – كتابة/سلسلة	يتم إعدادها في VBA. تم إعدادها علـــى اسم القائمة لعرض شريط قائمة مخصص من خلال التطبيق.

الجدول ٣-١ : خصائص كائن Application

الخاصية	نوع البيانات/الوصول	البيان
Shortcut Menu Bar	قراءة – كتابة/سلسلة	يتم إعدادها في VBA. تم إعدادها لعرض شريط قائمة اختصارات مخصص عام وذلك عند نقر نموذج أو تقرير نقرة يمنى.
User Control	قراءة – فقط/Boolean	يقوم أكسس بإعداد هذه الخاصية على صحيح/خاطئ في VBA. يتم استخدامها لتحديد ما إذا قام المستخدم ببدء تطبيق أكسس الحالي "صحيح" أم تام به تطبيق آخر يستخدم آلية OLE "خاطئ".

سوف يتغلب أي شريط قائمة مخصص أو شريط قائمة اختصار قمت بإعداده لنموذج أو عنصر تحكم على أي شريط قائمة مخصص أو شريط قائمة اختصار قمت بإعدادها من خلال خصائص الكائن Application مما يعني أنه إذا قمت بإعداد خاصية Menu Bar الخاصة بنموذج أو تقرير لشريط قائمة مختلف ويكون التركيز على هذا النموذج أو التقرير. عندئذ يتم عرض شريط القائمة المخصص بدلاً من شريط القائمة الذي تم إعداده كخاصية Menu Bar لكائن Application. إذا قمت بإعداد خاصية Shortcut – Menu Bar الخاصة بعناصر تحكم نموذج أو بنموذج أو بتقرير لشريط قائمة مختلف ويكون مؤشر الماوس على الكائن عند نقره نقرة يمنى، يتم عرض قائمة الاختصار المخصصة بدلاً من تلك التي تم إعدادها لكائن Application.

ملاحظة عند إعداد الخيارات Menu Bar وShortcut Menu Bar في الحوار Startup يتم ذلك إعداد الخصائص Startup Menu Bar وShortcut Menu Bar للكائن Application. يمكن الاختلاف في استخدام أكسس للخصائص التي قمت إعدادها في الحوار Startup عند بدء قاعدة البيانات. يمكنك إعداد خصائص Application المطابقة في إجراء VBA الذي يعمل بعد د قاعدة البيانات "ويتجاوز إعدادات الحوار Startup".

كائنات مجموعة نماذج وتقارير وصفحة الوصول للبيانات والتحكم

عند فتح قاعدة البيانات لأول مرة، يقوم أكسس بإنشاء ثلاثة مجموعات هي Forms وهي مجموعة كل النماذج المفتوحة، ومجموعة Reports وهي مجموعة كل التقارير المفتوحة و Data Access Page وهي مجموعة كل صفحات الوصول للبيانات المفتوحة. يقوم أكسس بتحديث كل مجموعة عند فتح وإغلاق النماذج والتقارير والصفحات الفردية. يحتوي كل نموذج وكل تقرير على كائن مجموعة Controls المحتوية على كل عناصر التحكم على النموذج أو التقرير. وتحتوي كل صفحة على كائن Web Application الذي يحتوي على سمات يستخدمها أكسس عند حفظ صفحة وصول إلى البيانات كصفحة ويب أو عند فتح صفحة ويب. تحتوي مجموعات Forms و Reports و Data Access Page و Controls على الخصائص الموضحة في الجدول ٣-٢.

الجدول ٣-٢: خصائص كائنات مجموعة Forms و Reports و Data Access Page و controls

الخاصية	نوع البيانات/الوصول	البيان
Application	قراءة فقط/كائن	يتم استخدامها للوصول إلى كائن Application النشط.
Count	قراءة فقط/عدد صحيح	يتم استخدامها لتحديد عدد النماذج والتقارير والصفحات وعناصر التحكم المفتوحة على نموذج أو تقرير.

كائنات نماذج وتقارير وصفحة الوصول للبيانات

يشير كائن Form إلى نموذج محدد مفتوح. كائنات Form هي من مجموعة Forms "إلا في حالة فتح أو إغلاق نموذج". يوجد أكثر من مائة خاصية لكائن Form تصف شكل النموذج وسلوكياته. يمكنك إعداد حوالي ٧٥ خاصية منها في صفحة النموذج. وتتضمن صفحة الخصائص أكثر من ٣٠ خاصية حدث يستطيع النموذج التعرف عليها.

يشير كائن Report إلى تقرير مفتوح محدد وتكون كائنات Report عضو من مجموعة Reports ولا يمكنك إضافة أو حذف كائن Report من مجموعة Reports "إلا عند فتح أو

إغلاق تقرير". يوجد أكثر من ١٠٠ خاصية لكائن Report تصف شكل التقرير وسلوكياته ويمكنك إعداد أكثر من ٥٠ خاصية في صفحة خصائص التقرير. تتضمن القائمة تسعة خصائص حدث يتعرف عليها التقرير.

يشير كائن Data Access Page إلى صفحة توصيل إلى البيانات محددة مفتوحة وتعتبر كائنات Data Access Page أعضاء من مجموعة Data Access Page. لا يمكنك إضافة أو حذف هذا الكائن في هذه المجموعة "إلا عن طريق فتح أو إغلاق صفحة". يوجد ١١ خاصية للكائن تصف شكل الصفحة وسلوكياتها، يمكنك إعداد ثلاثة خصائص منها وهي "Connection String وtag وVisible". ولكن لا تستطيع صفحات الوصول إلى البيانات التعرف على أي حدث لأنها لا تحتوي على وحدات.

يقوم الجدول ٣-٣ بسرد بعض خصائص الكائنات Form وReport وData Access Page والتي تكون مفيدة في برمجة VBA على وجه التحديد، ويكون أغلب هذه الخصائص متاحاً فقط في برمجة VBA وغير مسردة في صفحة الخصائص للنموذج أو التقرير أو صفحة الوصول إلى البيانات.

الجدول ٣-٣: مختارات من خصائص كائنات نماذج وتقارير وصفحة الوصول للبيانات

الخاصية	نوع البيانات/الوصول	متاح في	البيان
ActiveControl	قراءة فقط/كائن	VBA وماكرو	للنماذج فقط، يتم استخدامها لتحديد عنصر التحكم النشط على نموذج مفتوح.
Application	قراءة فقط/كائن	VBA	يتم استخدامها للوصول إلى كائن Application النشط.
Count	قراءة فقط/عدد صحيح	VBA وماكرو	يتم استخدامها لتحديد عدد العناصر الموجودة في مجموعة محددة.
CurrentRecord	قراءة فقط/طويل	VBA وماكرو	للنماذج فقط يتم استخدامها لتعريف السجل الحالي. تحتوي الخاصية على القيمة الطويلة الموضحة في مربع رقم السجل في أسفل الجانب الأيسر من النموذج.

الجدول ٣-٣: مختارات من خصائص كائنات نماذج وتقارير وصفحة الوصول للبيانات

الخاصية	نوع البيانات/الوصول	متاح في	البيان
CurrentView	قراءة فقط/عدد صحيح	VBA وماكرو	للنماذج وصفحات الوصول إلى البيانات فقط. يتم استخدامها لتحديد كيفية عرض نموذج أو صفحة حاليا. للخاصية القيمة صفر لأسلوب العرض Design والقيمة ١ لأسلوب العرض Form "يسمى أسلوب عرض Page عند الإشارة إلى صفحة الوصول إلى البيانات) والقيمة ٢ لأسلوب العرض Datasheet.
Cycle	قراءة-كتابة/بايت	VBA وماكرو وصفحة الخصائص	للنماذج فقط. يتم استخدامها لتحديد ما حدث عند الانتقال من آخر عنصر تحكم على نموذج لكل السجلات عامة والسجلات والصفحات الحالية.
Dirty	قراءة-فقط/Boolean	VBA وماكرو	للنماذج فقط.يتم استخدامها لتحديد ما إذا تم تعديل السجل الحالي منذ آخر حفظ له "صحيح" أم لا "خاطئ".
Filter	قراءة-فقط/سلسلة	VBA وماكرو وصفحة الخصائص	يتم استخدامها لتحديد مجموعة فرعية من السجلات ليتمم عرضها عند تطبيق التصفية للنموذج أو التقرير. وهي عبارة عن WHERE لصياغة SQL بدون استخدام الكلمة الأساسية WHERE، ويأخذ نموذج أو تقرير جديد هذه الخاصية لمصدر البيانات الذي تم إنشاؤه منه، ويتم حفظ التصفيات مع النموذج أو التقرير ولكن لا يتم تطبيقها آليا عند فتح النموذج.

الجدول ٣-٣: مختارات من خصائص كائنات نماذج وتقارير وصفحة الوصول للبيانات

الخاصية	نوع البيانات/الوصول	متاح في	البيان
FilterOn	قراءة-كتابة/Boolean	VBA وماكرو	يتم استخدامها لتحديد أو لتعيين ما إذا تم تطبيق خاصية Filter للنموذج أو التقرير "صحيح" أم لا "خاطئ".
Form	قراءة-فقط/كائن	VBA وماكرو	للنماذج فقط. يتم استخدامها للإشارة إلى كائن Form أو إلى النموذج النشط.
HasData	قراءة-فقط/عدد صحيح طويل	VBA وماكرو	للنماذج فقط. يتم استخدامها لتحديد ما إذا كان التقرير منضم إلى مجموعة سجلات فارغة. تكون القيمة -١ إذا كان للتقرير بيانات، أو تكون صفر إذا لم يكن هناك أي بيانات، أو تكون ١ إذا كان ١ إذا كان التقرير غير منضم.
HasModule	قراءة-كتابة/Boolean	VBA وماكرو وصفحة الخصائص	يتم إعداد هذه الخاصية بصورة افتراضية على لا حتى تقوم بأول عرض لوحدات النموذج أو التقرير. يسمى النموذج أو الجدول الذي لا يحتوي على وحدات باسم نموذج أو تقرير خفيف الوزن. يمكنك أعداد هذه الخاصية فقط لأسلوب عرض Design للنموذج أو التقرير.
HWnd	قراءة-فقط/طويل	VBA وماكرو	للنماذج فقط، يتم استخدامها لتحديد العدد الصحيح الطويل الفريد "يسمى معالج النافذة) والتي يقوم ويندوز بتخصيصها للنافذة الحالية.
KeyPreview	قراءة-كتابة/Boolean	VBA وماكرو وصفحة الخصائص	للنماذج فقط، يتم استخدامها لتحديد ما إذا استقبل النموذج أحداث لوحة المفاتيح قبل استقبال عنصر التحكم النشط لها "صحيح" أم لا "خاطئ".

الجدول ٣-٣: مختارات من خصائص كائنات نماذج وتقارير وصفحة الوصول للبيانات

الخاصية	نوع البيانات/الوصول	متاح في	البيان
MneuBar	قراءة-كتابة/سلسلة	VBA وماكرو وصفحة الخصائص	يتم إعدادها لاسم شريط القائمة المواد عرضه. ولعرض شريط القائمة المضمن باستخدام ماكرو أو VBA قم بإعداد هذه الخاصية لسلسلة طولها صفر (" "). لعرض نموذج بدون شريط قائمة، قم بإعداد الخاصية لقيمة ليست اسم لشريط قائمة موجود أو ماكرو لشريط قائمة.
Modal	قراءة-كتابة/Boolean	VBA وماكرو وصفحة الخصائص	للنماذج فقط، قم بإعداد هذه الخاصية على صحيح لتحديد أن كل نوافذ أكسس الأخرى غير معطلة عند فتح النموذج في أسلوب عرض Form من نافذة قاعدة البيانات أو ماكرو أو VBA أو عن طريق الانتقال من أسلوب عرض Design.
NewRecord	قراءة-فقط/Boolean	VBA وماكرو	للنماذج فقط يتم استخدامها لتحديد ما إذا كان السجل الحالي سجلا جديدا "صحيح" أم لا "خاطئ". عند الانتقال إلى سجل جديد والبدء في تحريره تبقي قيمة هذه الخاصية صحيحة حتى يتم حفظ التغييرات. يمكن عرض الخاصية فقط في أسلوب عرض Form أو Datasheet.

الجدول ٣-٣: مختارات من خصائص كائنات نماذج وتقارير وصفحة الوصول للبيانات

الخاصية	نوع البيانات/الوصول	متاح في	البيان
OrderBy	قراءة-كتابة/سلسلة	VBA وماكرو وصفحة الخصائص	يتم استخدامها لتحديد كيفية فرز السجلات في نموذج أو تقرير. قم بفصل الحقول بفاصلة، ولفرز حقل وفق ترتيب تنازلي اكتب DESC بعد السلسلة.
OrderByOn	قراءة-كتابة/Boolean	VBA وماكرو	يتم استخدامها لتحديد ما إذا تم تطبيق خاصية OrderOn "صحيح" أم لا "خاطئ"
Painting	قراءة-كتابة/Boolean	VBA وماكرو	يتم استخدامها لتحديد ما إذا تم إعادة طباعة نموذج أو تقرير "صحيح" أم لا "خاطئ"
Picture	قراءة-كتابة/نقطي	VBA وماكرو وصفحة الخصائص	يتم استخدامها لتحديد مسار النقطي ليتم عرضها كصورة خلفية على نموذج أو تقرير. يمكن استخدامها كذلك لعرض نقطي على زر أمر أو عنصر تحكم صورة أو زر تبديل أو عنصر تحكم تبويب.
PopUp	قراءة-كتابة/Boolean	VBA وماكرو وصفحة الخصائص	للنماذج فقط. قم بإعدادها على نعم لتحديد أن النموذج يبقى في أعلى نوافذ أكسس عند فتح النموذج في أسلوب عرض Form من إطار Database أو ماكرو أو VBA أو عن طريق الانتقال من أسلوب عرض Design. يمكنك إعداد هذه الخاصية فقط في أسلوب عرض Design للنموذج.

الجدول ٣-٣: مختارات من خصائص كائنات نماذج وتقارير وصفحة الوصول للبيانات

الخاصية	نوع البيانات/الوصول	متاح في	البيان
RecordsetC-lone	قراءة-فقط/كائن	VBA وماكرو	للنماذج فقط. يتم استخدامها للحصول على صلاحية الوصول إلى بعض خصائص مجموعة سجلات النموذج كما هو محدد بواسطة خاصية RecordSource مثل RecordCOUNT.
RecordSour-ce	قراءة-كتابة/سلسلة	VBA وماكرو وصفحة الخصائص	يتم استخدامها لتحديد مصدر البيانات لنموذج أو تقرير. ربما يكون مصدر البيانات جدولا أو استعلاما أو صياغة SQL
Report	قراءة-فقط/كائن	VBA وماكرو	للتقارير فقط. يتم استخدامها لكائن Report
Section	قراءة-فقط/عدد صحيح	VBA وماكرو	يتم استخدامها لتحديد مقطع أو عناصر تحكم في مقطع نموذج أو تقرير وتوفر صلاحية الوصول إلى خصائص المقطع
ShortcutMe-nuBar	قراءة-كتابة/سلسلة	VBA وماكرو وصفحة الخصائص	يتم إعدادها على قائمة الاختصارات المعروضة عند نقر نموذج أو عنصر تحكم نموذج أو تقرير نقرة يمنى. لعرض قائمة الاختصارات المضمنة، قم بإعداد السلسلة التي طولها صفر ("").

الجدول ٣-٣: مختارات من خصائص كائنات نماذج وتقارير وصفحة الوصول للبيانات

الخاصية	نوع البيانات/الوصول	متاح في	البيان
StatusBarT-ext	قراءة-كتابة/سلسلة	VBA وماكرو وصفحة الخصائص	للنماذج فقط. يتم استخدامها لتحديد النص المعروض في شريط المعلومات عند تحديد عنصر تحكم.
TabIndex	قراءة-كتابة/عدد صحيح	VBA وماكرو وصفحة الخصائص	للنماذج فقط. يتم استخدامها لتحديد مكان عنصر التحكم في ترتيب التبويب على نموذج.
TapStop	قراءة-كتابة/Boolean	VBA وماكرو وصفحة الخصائص	لعناصر التحكم على النماذج فقط. يتم استخدامها لتحديد ما إذا كان يمكنك استخدام مفتاح Tabلنقل التركيز إلى عنصر تحكم في أسلوب عرض Form "صحيح" أم لا "خاطئ".
Tag	قراءة-كتابة/سلسلة	VBA وماكرو وصفحة الخصائص	يتم استخدامها للتحكم في أي معلومات إضافية ترغب في تخزينها عن النماذج/التقارير/صفحات الوصول إلى البيانات أو المقاطع أو عناصر التحكم.
Toolbar	قراءة-كتابة/سلسلة	VBA وماكرو وصفحة الخصائص	يتم إعدادها على اسم شريط الأدوات المراد عرضه على نموذج أو تقريره. لعرض شريط الأدوات المضمن، اترك إعداد خاصية Toolbarفارغاً.
Visible	قراءة-	VBA وماكرو	يتم استخدامها لعرض/إخفاء

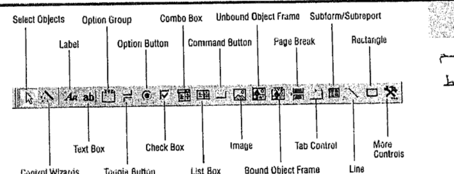
الجدول ٣-٣: مختارات من خصائص كائنات نماذج وتقارير وصفحة الوصول للبيانات

الخاصية	نوع البيانات/الوصول	متاح في	البيان
	كتابة/Boolean	نموذج/تقرير أو الوصول إلى البيانات أو عنصر تحكم.	

ملاحظة عند إعداد خاصية Modal على نعم يسمى النموذج نموذجا شرطيا وتبقى قوائم أكسس نشطة ويمكنك تنشيط نافذة في تطبيق آخر. ولتعطيل قوائم وشرائط أدوات أكسس ونوافذ تطبيقات أخرى، قم بإعداد كلا من الخاصيتين Modal وpopUp على نعم. بإعداد خاصية PopUp على نعم، يسمى النموذج بنموذج منبثق و لا يكون شريط أدوات النموذج نشطا.

كائن التحكم

يمثل كائن Object عنصر تحكم موجود على نموذج أو تقرير. تنتمي عناصر التحكم الموجودة على نموذج أو تقرير إلى مجموعة Controls لهذا النموذج أو التقرير. بالإضافة إلى عناصر التحكم المضمنة التي تظهر في مربع الأدوات "انظر الشكل ٣-٨"، يوجد أيضا عناصر تحكم رسم بياني مضمنة يمكنك إضافتها إلى مربع الأدوات. يمكنك كذلك استخدام عناصر تحكم مخصصة تسمى عناصر تحكم Active X لتزويد التطبيق بميزات إضافية "لمزيد من المعلومات عن عناصر التحكم المخصصة انظر الفصل ١٥". يمثل كائن Control كلا من عناصر التحكم المخصصة و المضمنة.



الشكل ٣-٨ عناصر التحكم المضمنة في شريط الأدوات.

لكل نوع من أنواع عناصر التحكم مجموعة خصائص متعلقة به تشمل خصـائص الأحـداث للأحداث التي يتعرف عليها عنصر التحكم. هناك خاصية معينة لبعض عناصر التحكـم وهـي الخاصية الافتراضية وهي أكثر الخصائص انتشارا لهذا النوع من عناصر التحكم. و الخاصيـة الافتراضية هي الخاصية التي يفترضها أكسس عند الإشارة إلى عنصر تحكم بدون تحيـد اسـم خاصية. عل سبيل المثال، عند الإشارة إلى مربع نص، يفترض أكسس أنك تشير إلـى خاصيـة Value.

عناصر تحكم Data هي عناصر التحكم التي يمكنها التحكم في البيانات ويمكـن أن تكـون مرتبطة بحقول الجدول. يحتوي عنصر تحكم البيانات على خاصية Control Source للإشـارة إلى مصدر البيانات. تتضمن عناصر تحكم البيانات السبعة على مربعات نص ومربعـات قابلـة للتحرير ومربعات قائمة وخانات اختيار وأزرار خيارات وأزرار تبديل ومجموعات خيار. ربمـا تكون عناصر التحكم الأخرى مرتبطة بكائنات أخرى كنموذج أو تقرير، وتحتوي عناصر التحكم هذه على خاصية Source Object للإشارة إلى الكائن أو على خاصية Source Doc للإشـارة إلى مصدر الملف المتضمن أو المرتبط. على سبيل المثال، تحتوي عناصر تحكم النموذج الفرعي أو التقرير الفرعي على خاصية Source Object التي يتم استخدامها لتحديد النموذج أو التقريـر المراد عرضه في عنصر التحكم. أما عناصر التحكم التي تعرض صورة تحتوي على خاصيـة Picture للإشارة إلى مصدر الصورة. على سبيل المثال، يمكـن لأزرار الأوامـر أو التبديـل عرض صور. يقوم الجدول ٣-٤ بسرد عناصر التحكم المضمنة وخصائصه الافتراضية وما إذا كان كل عنصر تحكم مرتبط بالبيانات أو بكائن آخر أو بملف أو بصورة.

الجدول ٣-٤ : عناصر التحكم المضمنة

عنصر التحكم	الخاصية الافتراضيـة	مرتبط بـ
إطار كائن منضم		SourceDoc
رسم بياني		SourceDoc
خانة اختيار	قيمة	ControlSource
مربع تحرير	نص	ControlSource
زر أمر		Picture

الجدول ٣-٤: عناصر التحكم المضمنة

عنصر التحكم	الخاصية الافتراضية	مرتبط بـ
إطار صورة		Picture
عنوان		
سطر		
مربع قائمة	نص	ControlSource
زر الخيار	قيمة	ControlSource
مجموعة الخيار	قيمة الخيار	ControlSource
فاصل الصفحة		
مستطيل		
نموذج فرعي/تقرير فرعي		SourceObject
عنصر تحكم التبويب		
مربع النص	قيمة	ControlSource
زر التبديل	قيمة	ControlSource و picture
إطار كائن غير منضم		SourceObject و sourceDoc

يحتوي كل نوع من أنواع عناصر التحكم على خصائصه المتعلقة به ولكن تشترك كل أنواع عناصر التحكم في نفس مجموعة الخصائص الأساسية. يقوم الجدول ٣-٥ بسرد خصائص كائن Control الأساسية.

الجدول ٣-٥: خصائص كائن التحكم فى جوهر أكسس

الخاصية	نوع البيانات/الوصول	متاح في	البيان
Application	قراءة-فقط/سلسلة	VBA	يتم استخدامها للوصول إلى كائن Application النشط لأكسس.
EventProcPrefix	قراءة-فقط/سلسلة	VBA وماكرو	يتم استخدامها لتحديد الجزء البادئ لاسم إجراء حدث.
InSelection	قراءة-كتابة/Boolean	VBA وماكرو	يتم استخدامها لتحديد ما إذا تم اختيار عنصر تحكم على نموذج في أسلوب عرض Design "صحيح" أم لا "خاطئ"
Left	قراءة-كتابة/عدد صحيح	VBA وماكرو وصفحة الخصائص	يتم استخدامها لتحديد مكان عنصر التحكم على نموذج أو تقرير. قيمة هذه الخاصية هي المسافة بوحدة القياس twips "وهي تساوي ١٤٤٠/١ من البوصة" من حد الأيسر لعنصر التحكم إلى الحافة اليسرى للمقطع المحتوي على عنصر التحكم.
Name	قراءة-كتابة/سلسلة	VBA وماكرو وصفحة الخصائص	يتم استخدامها لتحديد تعبير سلسلة يقوم بتعريف اسم عنصر التحكم.
Parent	قراءة-فقط/كائن	VBA وماكرو	يتم استخدامها للإشارة إلى أصل عنصر التحكم أو المقطع.
Section	قراءة-فقط/عدد صحيح	VBA وماكرو	يتم استخدامها لتحديد مقطع النموذج أو التقرير حيث يظهر عنصر التحكم. وتكون هذه الخاصية عدد صحيح مطابق لمقطع محدد.

الجدول ٣-٥ : خصائص كائن التحكم فى جوهر أكسس

الخاصية	نوع البيانات/الوصول	متاح في	البيان
Tag	قراءة –كتابة/سلسلة	VBA وماكرو وصفحة الخصائص	يتم استخدامها لتخزين معلومات إضافية عن عنصر التحكم. يمكنك استخدام هذه الخاصية في إنشاء خصائص المستخدم المعرفة الخاصة بك.
Top	قراءة – كتابة/عدد صحيح	VBA وماكرو وصفحة الخصائص	يتم استخدامها لتحديد مكان عنصر التحكم على نموذج أو تقرير.قيمة هذه الخاصية هي المسافة بوحدة القياس twips "وهي تساوي ١٤٤٠/١ من البوصة" من الحد الأعلى لعنصر التحكم إلى الحافة العليا للمقطع المحتوي على عنصر التحكم.
Visible	قراءة- كتابة/Boolean	VBA وماكرو وصفحة الخصائص	يتم استخدامها لإظهار أو إخفاء عنصر تحكم. "لا يمكن إعداد هذه الخاصية لعنصر التحكم Page-Break من خلال صفحة الخصائص". عند إعداد خاصية Visible لعنصر التحكم على لا، يكون التأثير تعطيل عنصر التحكم.

تقوم خاصية Parent بإرجاع كائن عنصر تحكم إذا كان الأصل عنصر تحكم وإرجاع كائن AccessObject إذا كان الأصل كائن مايكروسوفت أكسس. يعتبر كائن Access Object كائن أكسس محدد في المجموعات التاليـة: AllForms وAllReports وAllMacros وAllModules وallDataAccessPages AllTables وAllQueries وAllViews وAllStoredProcedures وAllDatabase Diagrams. وتقوم خاصية Parent لعنصر تحكم عنوان بإرجاع عنصر التحكم المرتبط به العنوان، وكذلك تقوم خاصية Parent لزر الخيار أو خانة الاختيار أو زر التبديل في مجموعة خيار بإرجاع عنصر تحكم مجموعة خيار.

تحتوي أغلب عناصر التحكم على خصائص إضافية متعددة، أقصاها عنصر التحكم Combo Box الذي يحتوي على أكثر من ٨٠ خاصية. يقوم الجدول ٣-٦ بسرد بعض خصائص عنصر التحكم تكون مهمة على وجه التحديد عند إنشاء برامج لجعل التطبيق آليا.

الجدول ٣-٦: خصائص إضافية مُحددة لكائنات Control

الخاصية	عنصر التحكم	نوع البيانات/الوصول	متاح في	البيان
BackColor	عنوان، مربع نص، مربع التحرير، مربع قائمة	قراءة-كتابة/طويل	VBA وماكرو وصفحة الخصائص	يتم استخدامها لتحديد لون لداخل عنصر تحكم أو مقطع. لاستخدام هذه الخاصية، يجب إعداد خاصية BackStyle على عادي "إن وجد".
BackStyle	عنوان، مربع نص، مربع التحرير	قراءة-كتابة/رقم أو سلسلة	VBA وماكرو وصفحة الخصائص	يتم استخدامها لتحديد هل سيكون عنصر التحكم واضحا. قم بإعدادها على ١ في VBA "أو على عادي في صفحة الخصائص" للإشارة إلى إعداد اللون الداخلي بخاصية BackColor أو على صفر "أو على واضح في صفحة الخصائص" للإشارة إلى أن عنصر التحكم واضحا.
Column	مربع التحرير،	قراءة-كتابة/عدد	VBA	يتم استخدامها

الجدول ٣-٦: خصائص إضافية مُحددة لكائنات Control

الخاصية	عنصر التحكم	نوع البيانات/الوصول	متاح في	البيان
	مربع نص	صحيح	وماكرو	للإشارة إلى عمود محدد أو توليفة عمود/صف، في مربع متعدد الأعمدة، يشير صفر إلى العمود أو الصف الأول ويشير ١ إلى العمود أو الصف الثاني وهكذا. على سبيل المثال، يشير العمود "٢،٠" إلى العمود الأول والصف الثالث.
ControlSou-rce	مربع النص، مربع التحريو، مربع القائمة، خانة الاختيار، زر التبديل، زر الخيار، مجموعة الخيار.	قراءة-كتابة/سلسلة	VBA وماكرو وصفحة الخصائص	يتم استخدامها لتحديد مصدر البيانات في عنصر تحكم. ولا تنطبق هذه الخاصية على عناصر تحكم خانة الاختيار أو زر الخيار أو زر التبديل في مجموعة خيار ولكنها تنطبق فقط على مجموعة الخيار نفسها.
Form	عنصر تحكم نموذج فرعي	قراءة-فقط/كائن	VBA وماكرو	يتم استخدامها للإشارة إلى النموذج المرتبط بعنصر تحكم نموذج فرعي.

الجدول ٣-٦: خصائص إضافية مُحددة لكائنات Control

الخاصية	عنصر التحكم	نوع البيانات/الوصول	متاح في	البيان
Height	مقطع النموذج و مقطع التقرير	قراءة-كتابة/رقم	VBA وماكرو وصفحة الخصائص	يتم استخدامها لتحجيم كائن لارتفاع محدد. تنطبق فقط على مقاطع النموذج ومقاطع التقرير وليس على النماذج أو التقارير.
HyperlinkAddress	زر الأمر، الصورة، العنوان	قراءة-كتابة/سلسلة	VBA وماكرو وصفحة الخصائص	يتم استخدامها لتحديد مسار كائن أو مستند أو عنوان بريد إلكتروني أو صفحة ويب التي هي هدف الارتباط التشعبي.
HyperlinkSubAddress	زر الأمر، الصورة، العنوان	قراءة-كتابة/سلسلة	VBA وماكرو وصفحة الخصائص	يتم استخدامها لتحديد مكان داخل الملف المستهدف المحدد في خاصية HyperlinkAddress. ولتحديد كائن في قاعدة البيانات الحالية، اترك خاصية HyperlinkAddress فارغة وحدد خاصية HyperlinkSubAddress باستخدام الصياغة object type

الجدول ٣-٦: خصائص إضافية مُحددة لكائنات Control

الخاصية	عنصر التحكم	نوع البيانات/الوصول	متاح في	البيان
				objectname
LimitToList	مربع التحرير	قراءة-كتابة/Boole-an	VBA وماكرو وصفحة الخصائص	يتم استخدامها لتحديد قيمة مربع التحرير للقيم التي تم سردها "صحيح" أم لا "خاطئ".
ListRows	مربع التحرير	قراءة-كتابة/عدد صحيح	VBA وماكرو وصفحة الخصائص	يتم استخدامها لإعداد أقصى عدد من الصفوف لعرضها في تعليق مربع القائمة لمربع التحرير
MultiSelect	مربع القائمة	قراءة-كتابة/بايت	VBA وماكرو وصفحة الخصائص	يتم استخدامها لتحديد ما إذا كان بإمكان المستخدم عمل عدة اختيارات في مربع القائمة
OldValue	مربع النص، مربع التحرير، مربع القائمة، خانة الاختيار، زر التبديل، زر الخيار، مجموعة الخيار.	قراءة-فقط/نفس نوع بيانات الحقل المرتبط بها عنصر التحكم.	VBA وماكرو	يتم استخدامها لتحديد القيمة غير المحررة في عنصر تحكم منضم. "عند حفظ التغييرات تصبح القيمة الحالية وOldValue نفس الشيء.

الجدول ٣-٦: خصائص إضافية مُحددة لكائنات Control

الخاصية	عنصر التحكم	نوع البيانات/الوصول	متاح في	البيان
Report	تقرير فرعي	قراءة-فقط/كائن	VBA وماكرو	يتم استخدامها للإشارة إلى التقرير المرتبط بعنصر تحكم تقرير فرعي.
RowSource	مربع التحرير، مربع القائمة، إطار كائن غير مرتبط، رسم بياني.	قراءة-كتابة/سلسلة	VBA وماكرو وصفحة الخصائص	يتم استخدامها لتحديد مصدر البيانات لعنصر التحكم.
RowSource-Type	مربع التحرير، مربع القائمة، الرسم البياني، إطار كائن غير منضم.	قراءة-كتابة/سلسلة	VBA وماكرو وصفحة الخصائص	يتم استخدامها لتحديد نوع البيانات التي تملأ القائمة.
SpecialEffe-ct	العنوان، مربع النص، زر الخيار، خانة الاختيار، مربع التحرير، مربع القائمة.	قراءة-كتابة/عدد صحيح أو سلسلة	VBA وصفحة البيانات وزر Special Effects "تأثيرات خاصة"	يتم استخدامها لتحديد ما إذا كان سيتم تطبيق تنسيقات خاصة لمقطع أو لعنصر تحكم. وتكون خيارات SpecialEffect هي Flat أو Raied أو Sunk-

الجدول ٣-٦: خصائص إضافية مُحددة لكائنات Control

الخاصية	عنصر التحكم	نوع البيانات/الوصول	متاح في	البيان
				en أو Etched أو Shadowed أو Chiseled. يتم إعداد هذه الخيارات في VBA على ٠ و ١ و ٢ و ٣ و ٤ و ٥ كل على حدة.
Text	مربع نص ومربع تحرير وسرد	قراءة-كتابة/سلسلة	VBA وماكرو	يتم استخدامها لتحديد البيانات المعروضة حاليا في عنصر التحكم عندما يكون التركيز على عنصر التحكم.
Transparent	زر الأمر	قراءة-كتابة/Boole-an	VBA وماكرو وصفحة البيانات	يتم استخدامها لتحديد ما إذا كان زر الأمر واضحا "صحيح" أم لا "خاطئ". باستخدام هذه الخاصية لا يتسم عرض الزر ولكنه يظل ممكنا. استخدم هذه

الجدول ٣-٦: خصائص إضافية مُحددة لكائنات Control

الخاصية	عنصر التحكم	نوع البيانات/الوصول	متاح في	البيان
				الخاصية لإنشاء عنصر تحكم أو مقطع نقطي يستجيب للنقر.
Value	خانة اختيار ومربع تحرير ومربع قائمة وزر خيار ومجموعة خيار ومربع نص وزر تبديل وعنصر تحكم تبويب.	قراءة-كتابة/مختلف	VBA وماكرو	يتم استخدامها لتحديد القيمة المحفوظة لخاصية عنصر التحكم الافتراضية.
Width	مقطع نموذج ومقطع تقرير	قراءة-كتابة/رقم	VBA وماكرو وصفحة الخصائص	يتم استخدامها لتحجيم كائن في العرض المحدد. تنطبق فقط على مقاطع النماذج ومقاطع التقارير وليس على النماذج والتقارير.

تستحق بعض خصائص عنصر التحكم التي تم سردها في جدول ٣-٦ مزيد من التوضيح. تقوم خاصية ControlSource بتحديد مصدر البيانات في عنصر تحكم. هناك ثلاثة احتمالات هي:

- ترك الخاصية فارغة. يكون عنصر التحكم غير منضم ويمكنك إعداد القيمة الخاصة به في ماكرو أو VBA.
- إدخال تعبير. يسمى عنصر التحكم باسم عنصر تحكم محتسب، يكون عنصر التحكم غير منضم ولا يمكن تغيير البيانات الموجودة فيه.
- تحديد اسم حقل في مجموعة السجلات الأساسية للنموذج. يسمى عنصر التحكم باسم عنصر تحكم منضم. عند تحرير البيانات في عنصر التحكم وحفظ التغييرات يمكنك تغيير القيمة المخزنة في الحقل.

وتكون خاصية Text متاحة للإعداد وللقراءة فقط عندما يكون التركيز كله على عنصر التحكم. قيمة خاصية Text هي البيانات المعروضة حالياً في عنصر التحكم عندما يكون التركيز عليه. وتختلف هذه الخاصية Value لعنصر التحكم وهي آخر بيانات تم حفظها لعنصر التحكم. عند الانتقال إلى عنصر تحكم آخر تكون خاصية Value مُعدة على البيانات الحالية. إذا قمت بحفظ السجل دون التنقل إلى عنصر تحكم آخر تكون إعدادات خاصية Text وValue نفس الشي. وبالنسبة لمربع التحرير يمكن أن تكون القيمة عنصر قائمة محدد أو سلسلة يمكنك الكتابة فيها. يمكنك استخدام خاصية Text لإرجاع النص أو لإعداده.

تحتوي خاصية Value على إعدادات مختلفة على حسب نوع عنصر التحكم:

- بالنسبة لخانة الاختيار أو زر الخيار أو زر التبديل، استخدم خاصية Value لتحديد ما إذا تم تحديد عنصر التحكم أم لا.
- بالنسبة لمربع النص، استخدم هذه الخاصية لتحديد قيمة عنصر التحكم المحفوظة ولإعداد القيمة لخاصية Text لعنصر التحكم وتكون البيانات سلسة.
- بالنسبة لمربع تحرير أو مربع قائمة أو مجموعة خيار. استخدم خاصية Value لتحديد أي قيمة أو خيار تم تحديده. يمكنك استخدام هذه الخاصية لإعداد مجموعة من مربعات التحرير لخاصية Text الخاصة بعنصر التحكم وإعداد مربع قائمة للقيمة الموجودة في العمود المنضم لعنصر القائمة المحدد وكذلك إعداد مجموعة خيار للإعداد Option Value لعنصر التحكم المحدد داخل المجموعة.
- بالنسبة لعنصر تحكم تبويب. تحتوي خاصية Value على رقم فهرس الصفحة الحالية.

كائن الشاشة

يشير كائن Screen إلى نموذج معين أو تقرير أو صفحة الوصول إلى البيانات أو عنصر تحكم والذي يكون عليه التركيز أو عنصر التحكم الذي كان عليه التركيز قبل ذلك. يمكنك باستخدام كائن Screen في إجراء VBA الإشارة إلى الكائن النشط دون معرفة اسم الكائن. ومع ذلك، الإشارة إلى كائن Screen لا يعني بالضرورة إلى جعل النموذج أو التقرير أو صفحة الوصول إلى البيانات أو عنصر التحكم كائناً نشطاً. يقوم جدول ٣-٧ بسرد خصائص كائن Screen. وتكون كل هذه الخصائص للقراءة فقط وتقوم بإرجاع مرجع إلى الكائن.

الجدول ٣-٧: خصائص كائن Object

الخاصية	البيان
Active Control	يتم استخدامها للإشارة إلى عنصر التحكم الذي عليه التركيز حالياً
Active Database	يتم استخدامها للإشارة إلى صفحة البيانات التي عليها التركيز.
Active Form	يتم استخدامها للإشارة إلى النموذج الذي عليه التركيز. وإذا كان التركيز على نموذجاً فرعياً، تشير هذه الخاصية إلى النموذج الرئيسي.
Active Report	يتم استخدامها للإشارة إلى التقرير الذي عليه التركيز.
Application	يتم استخدامها للإشارة إلى أكسس.
Previous Control	يتم استخدامها للإشارة إلى عنصر التحكم الذي كان عليه التركيز مسبقاً.

يشرح الفصل الثاني طريقة التحديد المركزية للبرمجة والتي تتطلب قيام إجراء VBA بتأسيس اتصال بكائن قبل اتخاذ أي إجراء عليه. يمكنك استخدام خصائص كائن Screen لعمل اتصال الكائن النشط.

نموذج كائن فيجوال بيسك لأكسس

يزود نموذج فيجوال بيسك لأكسس بثلاثة كائنات هم Debug و Err و Collection. يتم استخدام كائن Debug فقط في برمجة VBA.

يمكنك استخدام كائن Debug لإرسال أمر إلى نافذة محددة تسمى نافذة Immediate. يمكنك عرض هذه النافذة عندما تكون أي نافذة نشطة وذلك بضغط Ctrl+G "انظر الشكل ٣-٩". ويمكنك استخدام هذه النافذة كذلك كمنصة انطلاق لتقييم التعبيرات ولعرض وإعداد قيم الخصائص ولتشغيل الإجراءات.

ولا يحتوي كائن Debug على أي خصائص ولكن يحتوي على طريقة واحدة لطباعة النص في نافذة Immediate وهي Print. وها هي الجملة:

Debug. Print *outputlist*

حيث يكون *outputlist* تعبير رقمي أو سلسلة تعبير أو قائمة تعبيرات رقمية أو سلسلة تعبير مفصول بينهما بمسافات أو فواصل منقوطة. إذا قمت بحذف المعامل *outputlist*، يتم طباعة سطر فارغ. وعند العمل في نافذة Immediate، لا يوجد ضرورة للإشارة إلى كائن Debug بوضوح. يمكنك استخدام الجملة:

Print *outputlist*

وبصورة بديلة يمكنك استخدام علامة الاستفهام "؟" كاختصار لكلمة Print كالآتي:

? *outputlist*

ستقوم باستخدام نافذة Immediate لاختبار بعض مراجع الخصائص والكائنات التي تم مناقشتها في المقطع التالي.

الشكل ٣-٩

نافذة Immediate.

الإشارة إلى الكائنات والخصائص حسب الاسم

قبل استخدام كائن في إجراء VBA، يجب عليك تحديد الكائن باستخدام قواعد قبل أكسس للإشارة إلى الكائنات. بالرغم من إمكانية استخدامك لمقاييس تسمية لتعريف وتوثيق الكائنات التي تقوم بإنشائها. إلا أن لأكسس قواعده الخاصة به للإشارة إلى الكائنات التي يجب أن تتبعها عند كتابة البرامج.

ملاحظة

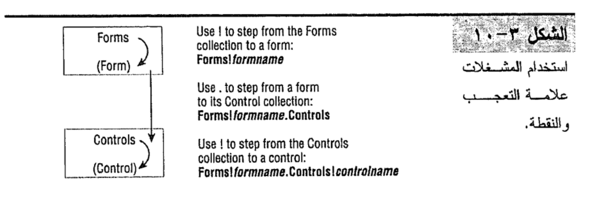
سيمكنك استخدام نافذة Immediate لاختبار وتقييم المراجع ولذلك سترغب في أن تكون أمام جهاز الكمبيوتر للعمل من خلال هذا المقطع استخدام تطبيق Expenses الذي تم إنشاؤه في الفصل ١. إذا لم تكن عملت خلال الفصل ١، اتبع الآن الخطوات في هذا الفصل لإنشاء قاعدة بيانات Expenses.

الإشارة إلى كائن بواسطة الاسم

الطريقة الوحيدة التي يمكن من خلالها الإشارة إلى كائن هي البدء بأعلى كائن في التدرج والانتقال من خلال مسار التدرج إلى الكائن وتسجيل أسماء الكائنات المُحددة وكائنات المجموعة التي تواجهها أثناء الانتقال في المسار. يمكنك استخدام علامة التعجب "!" والمشغل والنقطة "." للتفرقة بين الخطوات والكائنات والمجموعات كالآتي:

- استخدم مشغل علامة التعجب عند الخروج من مجموعة إلى أحد أعضائها في الصيغة *collectionname.objectname*.
- استخدم مشغل النقطة عند الخروج من كائن إلى إحدى مجموعات في الصيغة *objectname. Collectionname*.

يوضح الشكل ٣-١٠ كيفية عمل المشغلات علامة التعجب والنقطة.



الشكل ٣-١٠ استخدام المشغلات علامة التعجب والنقطة.

الإشارة إلى نموذج أو تقرير

للإشارة إلى نموذج مفتوح. على سبيل المثال، نموذج Switchboard في تطبيق Expenses. ابدأ بالكائن Application وانتقل إلى مجموعة Forms ثم إلى نموذج Switchboard في المجموعة:

.Application.Forms!Switchboard

للإشارة إلى تقرير مفتوح. على سبيل المثال، تقرير Expense Report. ابدأ بكائن Application وانتقل إلى مجموعة Reports ثم إلى تقرير Expense Report في المجموعة:

.Application.Reports!Expense Report

عندما يحتوي اسم كائن على مسافات، يجب عليك وضع الاسم في أقواس مربعة. وإلا يمكنك حذف الأقواس المربعة "سوف يقوم أكسس بإدخال الأقواس المربعة لك".

يمكن تقليل طول مرجع باستخدام الافتراضات. على سبيل المثال، يفترض أكسس أنك موجود في تطبيق أكسس عند الإشارة إلى الكائنات مما يعني أنك لست في حاجة إلى الإشارة لكائن Application بوضوح وتصبح المراجع كالآتي:

Forms!Switchboard

Report!{Expense Report}

وتبقى هذه المراجع مسارات كاملة للمراجع تشير إلى نموذج أو تقرير مُحدد بواسطة الاسم.

الإشارة إلى خصائص النموذج والتقرير

يمكنك استخدام مشغل النقطة للفصل بين كائن وخاصية كائن في النموذج *objectname.propertyname*. على سبيل المثال، للإشارة إلى خاصية RecordSource لنموذج Expense Categories. قم باستخدام هذا المرجع:

Forms!{Expenses Categories}.RecordSource

ملاحظة عندما يحتوي اسم خاصية على أكثر من كلمة، تقوم صفحة الخصائص بعرض مسافات بين الكلمات. على سبيل المثال، تعرض صفحة الخصائص لنموذج اسم خاصية RecordSource كـ Record Source. يجب عليك حذف المسافات عند إنشاء مرجع لخاصية.

استخدام نافذة Immediate لتقييم خاصية كائن

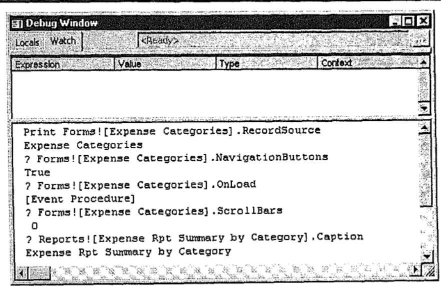
يمكنك استخدام خاصية Immediate لتحديد إعداد خاصية كائن. لفتح نافذة Immediate، ببساطة اضغط Ctrl+G. وبفتح نموذج أو تقرير اكتب Print أو ? يليها مرجع الخاصية المراد تقييمه ثم اضغط Enter. يقوم أكسس مباشرةً بتقييم مرجع الخاصية وعرض قيمة إعداد الخاصية في السطر التالي لنافذة Immediate.

تقوم نافذة Immediate بتنفيذ سطر واحد في كل مرة تضغط فيها Enter. يمكنك استخدام عدة أوامر مألوفة لتحرير نص في نافذة Immediate منها أمر "قص" Cut و"نسخ" Copy و"لصق" Paste و"حذف" Delete و"تحديد الكل" Select All في قائمة Edit. يمكنك تحرير سطر قمت بتنفيذه بالفعل ثم ضغط Enter لتنفيذ السطر الذي تم تحريره "يُضيف أكسس سطر جديد أسفل السطر المُحرر ويعرض نتيجة التنفيذ".

يوضح الشكل ٣-١١ مثالاً لتطبيق Expenses، لاحظ أنه عندما تكون القيمة نعم/لا مثل قيمة خاصية Navigation Button يقوم أكسس بتحويل نعم إلى صحيح و لا إلى خاطئ. وكذلك، إذا لم يتم إعداد قيمة للخاصية، يقوم أكسس بعرض سطر فارغ.

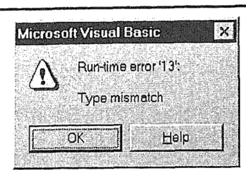
الشكل ٣-١١

استخدام نافذة Immediate لتقييم إعداد خاصية.



إذا كانت الخاصية مما يمكن إعدادها في أسلوب عرض Design "خاصية تصميم وقتي". بالتالي يمكن فتح النموذج أو التقرير في أي أسلوب عرض عند تقييم الخاصية. وإذا كان للخاصية قيمة يتم تحديدها فقط عندما يكون النموذج أو التقرير في طور التشغيل مثل خاصية Dirty. في هذه الحالة يجب أن يكون النموذج أو التقرير في طور التشغيل وإلا سيعرض أكسس رسالة خطأ.

تستطيع نافذة Immediate عرض قيم النص فقط، إذا قمت بكتابة مرجع لنموذج مفتوح. مثل: ?Forms!{Expense Reports by Employee} ثم قمت بضغط Enter. سيعرض أكسس رسالة خطأ "انظر الشكل ٣-١٢". وإذا كان النموذج أو التقرير مغلقاً عندما تحاول تقييم خاصية مثل: ?Forms!{Expense Categories}.RecordSource يقوم أكسس بعرض نفس رسالة الخطأ.



الشكل ٣-١٢
رسالة الخطأ التي ظهرت عند محاولة تقييم مرجع لكائن أو مرجع لخاصية متعلقة بنموذج أو تقرير مغلق.

الإشارة إلى عنصر تحكم

للإشارة إلى عنصر تحكم على نموذج مفتوح. ابدأ من أعلى التدرج بالكائن Application وانتقل إلى مجموعة Forms ثم إلى النموذج المحدد ومن ثم إلى مجموعة Controls وأخيراً إلى عنصر التحكم:

Forms!Switchboard.Controls!*controlname*

على سبيل المثال، للإشارة إلى زر الأمر Option 1 على Switchboard انتقل خلال المسار أولاً إلى مجموعة Controls ثم إلى عنصر التحكم المحدد:

Forms!Switchboard.Controls!Option1

الإشارة إلى خصائص عنصر التحكم

للإشارة إلى خاصية عنصر التحكم. اجعل النقطة واسم الخاصية معلقة إلى نهاية مرجع عنصر التحكم كالآتي:

Forms!Formname!Controlname!propertyname

على سبيل المثال، للإشارة إلى البيانات المحفوظة في عنصر تحكم مربع النص Expense Categories على نموذج Expense Categories، استخدم خاصية Value:

Forms!{Expense Categories}!Expensecategory.Value

يمكن أن يكون للكائن خاصية افتراضية يفترضها أكسس عند عدم تحديد اسم خاصية بوضوح. وتكون خاصية Value هي الخاصية الافتراضية لعنصر تحكم مربع نص. وباستخدام الافتراضيات يكون المرجع للبيانات المحفوظة في مربع النص هو:

Forms!{Expense Categories}!Expensecategory

تم مناقشة خاصية Value وخاصية Text لكائن Control سابقاً في هذا الفصل. ولاكتشاف خصائص Value وText:

١ - افتح نموذج Expense Category في أسلوب عرض Forms.

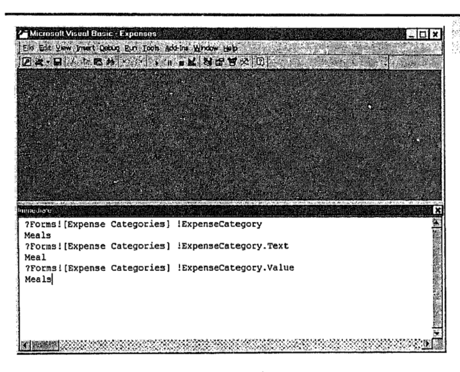
٢ - قم بتغيير Expense Category للسجل الأول بـ Mail واضغط Enter. ثم انقر حقل Expense Category، ولإظهار نافذة Immediate اضغط Ctrl+G.

٣ - اكتب كل سطر مما يلي بدون حفظ السجل واضغط Enter لتقييم التعبير "انظر الشكل ٣-١٣".

Forms!{Expense Categories}!Expensecategory

Forms!{Expense Categories}!Expense Category.Text

Forms!{Expense Categories}!Expense Category.Value



الشكل ٣-١٣ اكتشاف خاصية Text وخاصية Value لعنصر تحكم مربع نص.

الخصائص التي تمثل كائنات أخرى

تحتوي أغلب الخصائص على قيمة نص تمثل إعدادها. يمكنك عرض هذه القيمة في نافذة Immediate. تحتوي بعض الكائنات على خصائص مُحددة يمكنك استخدامها للإشارة إلى كائن آخر. على سبيل المثال، تشير خاصية Parent لعنصر تحكم إلى الكائن الأصلي لعنصر التحكم. وتشير خاصية Parent لمربع نص إلى النموذج الذي يحتوي على مربع النص. يقوم الجدول ٣-٨ بسرد الكائنات المحتوية على خصائص تشير إلى كائنات أخرى. "يتضمن هذا الجدول

خاصة Me وخاصية RecordsertClone الموجودة فقط في VBA. ستعرف المزيد عن هذه الخصائص في الفصل ٥".

الجدول ٣-٨: خصائص الكائنات التي تشير لكائنات أخرى

الكائن	خاصية الكائن	تشير إلى
Screen	Active Form	النموذج المحتوى على عنصر التحكم الذي عليه التركيز أو النموذج الذي عليه التركيز.
Screen	Active Data Sheet	صفحة البيانات التي عليها التركيز.
Screen	Active Report	التقرير الذي عليه التركيز.
Screen	Active Control	عنصر التحكم الذي علية التركيز.
Screen	Previous Control	عنصر التحكم الذي كان عليه التركيز قبل عنصر التحكم الموجود على نفس النموذج والذي عليه التركيز حالياً.
Control	Parent	الكائن الأصلي لعنصر التحكم الأصلي.
Subform Control	Form	النموذج المعروض داخل عنصر تحكم نموذج فرعي.
Subreport Control	Report	التقرير المعروض في عنصر تحكم نموذج فرعي.
Form	Form	النموذج نفسه.
Form	Me	النموذج نفسه.
Form	Recordset Clone	مجموعة السجلات.

الجدول ٣-٨: خصائص الكائنات التي تشير لكائنات أخرى

الكائن	خاصية الكائن	تشير إلى
Report	Report	التقرير نفسه.
Report	Me	التقرير نفسه.

يمكنك اختبار هذه المراجع في نافذة Immediate لأن هذه الخصائص تشير إلى كائن وليس إلى قيمة. على سبيل المثال، يمكنك استخدام خاصية Parent لمربع نص للإشارة إلى النموذج نفسه. ولكن إذا كتبت عبارة ?Forms!{Expense Categories}!Expense Category.Parent واضغط Enter. يقوم أكسس بعرض رسالة خطأ.

الإشارة إلى نموذج فرعي

الطريقة الشائعة لعرض البيانات من جدولين هي إنشاء نماذج على أساس كل جدول من الجداول ووضع عنصر تحكم لنموذج فرعي على نموذج واحد من النماذج وعرض النموذج الثاني داخل عنصر تحكم النموذج الفرعي. وبهذا الترتيب يسمى النموذج المحتوى الفرعي باسم النموذج الرئيسي، بينما يسمى النموذج المعروض في عنصر تحكم النموذج الفرعي باسم النموذج الفرعي.

لاكتشاف المراجع لنموذج فرعي، قم بالآتي:

١- افتح Expense Reports بواسطة نموذج Employee في أسلوب عرض Form. تكون خاصية Name لعنصر التحكم الفرعي هي Employee Subform وبذلك يمكنك الإشارة إلى عنصر تحكم النموذج الفرعي باستخدام المرجع Forms!{Expense Reports by Employee}!{ Employees Subform}.

٢- يمكنك تقييم خصائص عنصر تحكم النموذج الفرعي في نافذة Immediate. على سبيل المثال، سنستخدم خاصية Source Object لتحديد اسم النموذج المعروض في عنصر تحكم النموذج الفرعي. اكتب: ?Forms!{Expense Reports by Employee}!{Employees Subform}.Source Object واضغط Enter.

٣- يمكنك الإشارة إلى النموذج المعروض في عنصر تحكم النموذج الفرعي باستخدام خاصية Form لعنصر تحكم النموذج الفرعي كالآتي: ?Forms!{Expense Reports by Employee}!{Employees Subform}.Form. يمكنك تقييم خصائص هذا النموذج في نافذة Immediate. سنقوم بتقييم خاصية Default View. اكتب:

?Forms!{Expense Reports by Employee}!{Employees Subform}.Form
.Default View. واضغط Enter. وسوف يقوم أكسس بعرض الرقم الصحيح ٢ الذي يمثل أسلوب عرض Datasheet.

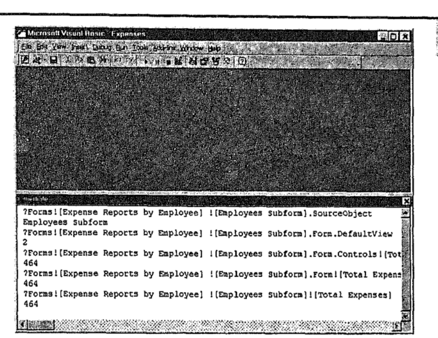
٤- يمكنك الإشارة إلى عنصر تحكم على نموذج معروض في عنصر تحكم نموذج فرعي بالإشارة أولاً إلى النموذج ثم الانتقال إلى مجموعة Controls ثم إلى عنصر التحكم المحدد. سوف نقوم بالإشارة إلى القيمة في عنصر التحكم Tools Expense على النموذج الفرعي. اكتب: ?Forms!{Expense Reports by Employee}!{Employees Subform}.Form!{Total Expenses} واضغط Enter.

٥- لحسن الحظ. يمكنك استخدام الافتراضيات لتبسيط المرجع لعنصر تحكم على نموذج فرعي. تعتبر مجموعة Collection هي المجموعة الافتراضية للنموذج الفرعي وبذلك يكون التبسيط الأول هو حذف المرجع لمجموعة Controls. اكتب:
?Forms!{Expense Reports by Employee}!{Employees Subform}.Form!{Total Expenses} واضغط Enter.

٦- يتعامل أكسس مع خاصية Form على أنها الخاصية الافتراضية لعنصر تحكم النموذج الفرعي عند الإشارة إلى عنصر تحكم على النموذج الفرعي. ولذلك، يمكنك حذف المرجع لخاصية Form. اكتب: ?Forms!{Expense Reports by Employee}!{Employees Subform}!{Total Expenses} واضغط Enter.

والجملة العامة للإشارة إلى عنصر تحكم على نموذج فرعي هي:
Form!forname!Subformcontrolname!controlname

يوضح الشكل ٣-١٤ نتائج اختيار هذه المراجع في نافذة Immedaite.



الشكل ٣-١٤

اختبار المراجع لعنصر تحكم نموذج فرعي ولنموذج معروض في عنصر تحكم نموذج فرعي في نافذة Immediate.

الإشارة إلى عناصر تحكم على النموذج أو التقرير النشط

تعتبر المراجع أو المعرفات التي قمنا باستكشافها مراجع بمسارات كاملة تم الحصول عليها عن طريق البدء من أعلى تدرج الكائن وعبور المسار للوصول إلى الكائن. وقمنا كذلك بتقصير المراجع عن طريق الرجوع إلى كائن Application بشكل مضمن واستخدام المجموعات والخصائص الافتراضية. يمكنك كذلك تقصير المرجع عندما تريد الإشارة إلى عنصر تحكم على النموذج أو التقرير النشط. يمكنك الإشارة إلى النموذج النشط بوضوح لأن أكسس يعرف أي نموذج هو نموذجاً نشطاً. بمعنى آخر، يمكنك تحديد عنصر تحكم مربع النص First Name ببساطة باستخدام First Name، يمكنك تحديد عنصر التحكم Total Expenses المعروضة في عنصر التحكم Employee Subform باستخدام {Employee Subform}!{Total Expenses}.

يسمى المحدد الذي يشير بوضوح إلى النموذج أو التقرير النشط باسم الصياغة القصيرة أو المرجع غير المؤهل. بينما يسمى المرجع الذي يتضمن مسار تدرجي كامل "ويستخدم الافتراضات ومرجع ضمني لكائن Application" باسم مرجع مؤهل كلية. وبالطبع، يمكنك استخدام المرجع المؤهل كلية بلا مشاكل ولكن هناك استثناءات عندما يتحتم عليك استخدام الجملة القصيرة بدلاً منه. تم مناقشة هذا بمزيد من التفاصيل في الفصول التالية.

لا يمكنك اختبار المحددات التي تشير إلى الكائن النشط عندما تعمل في نافذة Immediate وذلك لأن نافذة Immediate هي النافذة النشطة، إذا حاولت اختيار مرجع غير مؤهل في نافذة Immediate، يقوم أكسس بعرض "خطأ مترجم: لم يتم تحديد اسم خارجي" رسالة.

استخدم كائن الشاشة للإشارة إلى الكائن النشط

لدى أكسس طريقة فريدة لتحديد النموذج أو التقرير أو عنصر التحكم النشط أو حتى عنصر التحكم الأخير الذي كان عليه التركيز دون استخدام الأسماء المحددة التي أعطيتها للكائن. يعتبر تجنب أسماء محددة شيئاً ضرورياً عند إنشاء كائنات ترغب في إعادة استخدامها في التطبيق الخاص بك. يمكنك استخدام خصائص الكائن Screen لتحديد الكائن النشط. على سبيل المثال، للإشارة إلى كائن RecordSource للنموذج النشط. استخدم المرجع Screen.ActiveForm.RscordSource. للإشارة إلى خاصية Locked لعنصر تحكم يسمى Last Name على النموذج النشط، قم استخدام المرجع Screen.ActiveForm.Last Name.Locked.

بالإضافة إلى ذلك، يمكنك الإشارة إلى عنصر التحكم النشط باستخدام الكائن Screen. على سبيل المثال، للإشارة إلى اسم عنصر التحكم النشط، يمكنك استخدام المرجع Screen.ActiveControl.Name. وكمثال آخر، يمكنك الإشارة إلى خاصية TabIndex لعنصر التحكم الموجود على النموذج النشط والذي كان التركيز عليه سابقاً باستخدام المرجع Screen.PreviousForm.TabIndex. يمكنك استخدام الكائن Screen في عناصر التحكم المحتسبة على النماذج وعلى إجراءات VBA للإشارة إلى كائن بدون تسميته بوضوح.

الإشارة إلى حقل

للإشارة إلى حقل عادة في جدول أو استعلام. وتعتمد جملة المرجع على ما إذا كان الجدول أو الاستعلام في مصدر السجل الأساسي للنموذج المفتوح.

الإشارة إلى حقل في سجل النموذج

يمكنك الإشارة إلى حقل في الجدول أو الاستعلام الذي هو مصدر السجل للنموذج سواء كان الحقل منضم لعنصر تحكم على النموذج باستخدام المرجع أم لا.

Forms!*formname!fieldname*

على سبيل المثال، قم إضافة الحقل DateHired إلى الجدول Employee وأدخل تواريخ توظيف نموذجية لكل موظف. "لا تقم إضافة عنصر تحكم إلى نموذج Expense Reports by Employee" ثم اكتب ?Forms!{Expense Reports by Employee}!DateHired. ثم اضغط Enter. وستعرض نافذة Immediate تاريخ التوظيف النموذجي للموظف.

ملاحظــة بصورة افتراضية. عند إنشاء نموذج باستخدام معالج نموذج تكون خاصية Name لكل عنصر تحكم تم إنشاؤه هي نفس اسم الحقل المرتبط به عنصر التحكم. عند إضافة عنصر تحكم إلى نموذج وذلك سحب حقل من قائمة الحقل. يأخذ عنصر التحكم اسم الحقل. ومع ذلك، يجب أن يكون اسم عنصر التحكم هو نفس اسم الحقل.

الإشارة إلى حقل في جدول أو استعلام

عند تصميم استعلام أو جملة SQL يجب عليك الإشارة إلى حقل في جدول أو استعلام. في كل حالة، ينتمي كائن Field إلى مجموعة Fields للجدول أو الاستعلام وتكون مجموعة Fields هي المجموعة الافتراضية للجدول والاستعلامات. لذلك يمكنك استخدام أي من نماذج الصياغة الآتية لحقل جدول وهما:

tablename.Fields!fieldname

tablename!fieldname

وبالنسبة لحقل استعلام. يمكنك استخدام إحدى نماذج الصياغة هذه:

queryname.Fields!fieldname

queryname!fieldname

ومع ذلك، يقوم محرك قاعدة البيانات Jet إدارة الجدول والاستعلامات في قاعدة البيانات أكسس "يقوم MSDE إدارة الجداول في مشروع. ويتم استبدال الاستعلامات بأساليب عرض وإجراءات مُخزنة، ويقوم MSDE بمعالجة كلاً منهما". يستخدم Jet عامل التشغيل النقطة أو علامة التعجب عند الانتقال من مجموعة أحد أعضائها. على سبيل المثال، يمكنك استخدام Employee!LastName أو Employee.LastName للإشارة إلى حقل LastName الموجود في جدول Employee.

لا يمكنك اختبار مراجع جدول أو استعلام مباشرةً في نافذة Immediate. عند العمل في نافذة Immediate. يمكنك استخدام وظيفة DLookup() لاختبار مرجع إلى حقل في جدول أو استعلام. على سبيل المثال، لاشتقاق القيمة الأولى في الحقل LastName في الجدول Employee. اكتب ?DLookup("LastName","Employees") في نافذة Immediate ثم اضغط Enter وسوف يقوم أكسس بعرض Davolio. لمزيد من المعلومات عن وظيفة DLookup. انظر الفصل ٤، سوف تتعلم في الفصل ٦ كيفية استخدام كائنات الوصول إلى البيانات لاستعادة البيانات من الجداول والاستعلامات.

استخدام التعبير في "منشئ التعبيرات" لإنشاء مراجع

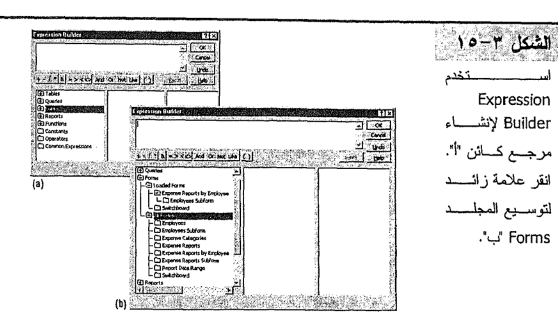
يعتبر استخدام التعبيرات للإشارة إلى الخصائص وعناصر التحكم على النماذج أو النماذج الفرعية أمر مُعقد، ولكن لحسن الحظ يقدم أكسس التعبير Expression Builder للمساعدة في إنشاء تعبيرات من أي نوع، منها تعبيرات لمعايير أو استعلام وإعدادات خاصية وكذلك لبرمجة VBA. ولكن لسوء الحظ، لا يكون Expression Builder متاحاً في نافذة Immediate.

يمكنك بدء Expression Builder بعدة طُرق:

- نقر الزر الأيمن للماوس فوق المكان الذي تريد فيه التعبير. اختر أمر Build Event من قائمة الاختصارات ثم انقر Expression Builder.
- انقر المكان الذي تريد فيه التعبير ثم الزر Build في شريط الأدوات.
- عند إنشاء تعبير في مربع تحرير خاصية أو في مربع تحرير معامل، يمكنك استدعاء Expression Builder بنقر زر Build الذي يظهر على يمين مربع التحرير.

في الحوار Expression Builder يحتوي مربع القائمة الموجود على اليسار على مجلدات لكل الجداول والاستعلامات والنماذج والتقارير في قاعدة بياناتك. وكذلك مجلدات للوظائف المُضمنة والثوابت وعوامل التشغيل والتعبيرات العامة ووظائف فيجوال بيسك المخصصة "انظر الشكل ٣-١٥ أ". ويكون Expression Builder حساساً للسياق. تعتمد مجموعة المجلدات التي تظهر في مربع القائمة الأول على المكان الذي بدأت منه الباني. في الشكل ٣-١٥ بدأ الباني من إحدى نوافذ البرمجة "نافذة Macro أو Module". يكون للمجلدات المحتوية على مجلدات أخرى علامة زائد. عند النقر لتوسيع المجلد تتحول علامة زائد لتصبح علامة ناقص.

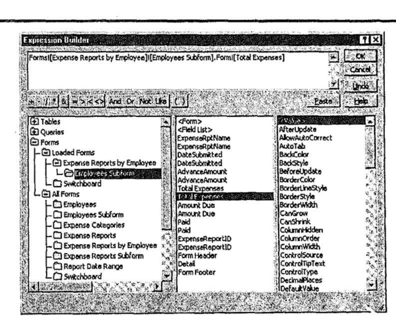
تحتوي مجلدات Forms أو Reports على مجلدات لكل نموذج وتقرير خاص بك وعلى مجلدات منفصلة للنماذج المفتوحة "في المجلد الفرعي "Loaded Forms". إذا كان هناك نموذج يحتوي على نموذج فرعي وكان مفتوحاً عند بدء Expression Builder، يتعرف أكسس على العلاقة بين النموذج والنموذج الفرعي ويقوم عرض مجلد للنموذج الفرعي داخل مجلد النموذج. في الشكل ٣-١٥ ب تم توسيع مجلد Forms كلية. يوضح الشكل أن نموذج Switchboard ونموذج Expense Reports by Employee هما النموذجان الوحيدان المفتوحان. وأن النموذج Expression Builder تعرف على Employees Subforms على أنه نموذج فرعي للنموذج Expense Report by Employee.



الشكل ٣-١٥

استخدم Expression Builder لإنشاء مرجع كائن "أ". انقر علامة زائد لتوسيع المجلد Forms "ب".

عند تحديد كائن مُحدد في مرع القائمة على اليسار، يتغير مربع القائمة الموجود في الوسط ليظهر الكائنات المتضمنة في الكائن المُحدد. عند تحديد نموذج، يكون العنصر الأول في القائمة الوسطى هو <Forms>، والذي يعرض النموذج نفسه. ويكون العنصر الثاني هو <Field List> والذي عرض قائمة الحقل نفسه للجدول أو الاستعلام الذي شكل النموذج. بينما تكون العناصر الأخرى المتقية هي عناصر التحكم والمقاطع على النموذج. عند تحديد عنصر في مربع القائمة الأوسط، يتغير مربع القائمة الموجود على الجانب الأيمن ليعرض خصائص عنصر تم تحديده. يوضح الشكل ٣-١٦ الاختبارات لمرع النص Total Expenses على نموذج Employees Subform. يتطابق الاسم المتكرر في مربع القائمة مع عنوان ومربع النص المرتبط ه والذي يحمل نفس الاسم على هذا النموذج.



الشكل ٣-١٦

المرجع المؤهل لعنصر تحكم على نموذج فرعي.

بعد الانتهاء من تحديد اختيارك، انقر الزر Paste وليقوم Expression Builder بإنشاء المرجع على أساس الاختيارات والسياق حيث بدأت الباني ويقوم بلصق المرجع في مربع التحرير في أعلى الحوار. يوضح الشكل ٣-١٦ المرجع المؤهل لمربع النص Total Expenses "لاحظ أن Expression Builder" يتضمن مرجع Form الافتراضي. يمكنك تحرير المرجع في مربع التحرير باستخدام لوحة المفاتيح وأزرار التشغيل في الحوار Builder. في هذا المثال، يمكنك تحرير المرجع لحذف المرجع Form أو يمكنك اختصار المرجع للمرجع غير المؤهل.

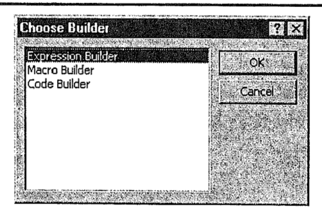
ولتوضيح كيفية تأثير مكان البداية على Expression Builder وانقر OK. لتبدأ الباني بالنموذج Expense Reports by Employee على أنه الكائن النشط:



١- مع نموذج Expense Reports by Employee في أسلوب عرض Design، حدد النموذج وانقر زر Build في شريط الأدوات. يعرض أكسس الحوار Choose Builder "انظر الشكل ٣-١٧".

الشكل ٣-١٧

الحوار Choose Builder.

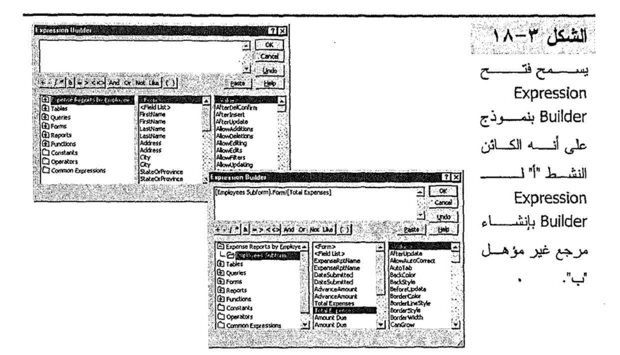


٢- اختر Expression Builder وانقر OK. يعرض Expression Builder مجلد للنموذج كالمجلد الأول في مربع القائمة الموجود على اليسار ويقوم بملء مربع القائمة في الوسط بعناصر التحكم لملء النموذج. "انظر الشكل ٣-١٨ أ"

٣- قم توسيع المجلد Expense Reports by Employee واختر Employee Subform. حدد Total Expenses في وسط مربع القائمة وحدد <Value> على يمين مربع القائمة. ثم انقر الزر Paste وسوف يقوم Expression Builder بلصق المرجع القصير لعنصر التحكم "انظر الشكل ٣-١٨ ب".

الشكل ٣-١٨

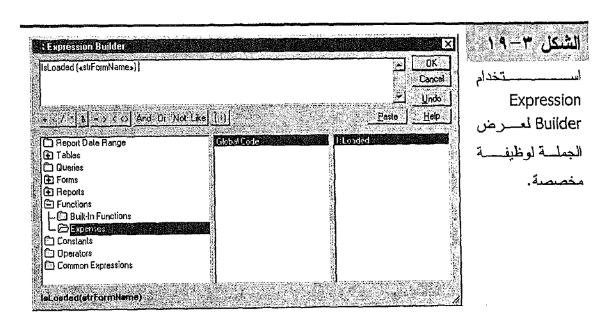
يسمح فتح Expression Builder بنموذج على أنه الكائن النشط "أ" لـ Expression Builder بإنشاء مرجع غير مؤهل "ب".



يمكنك كذلك استخدام Expression Builder لإنشاء التعبيرات متضمنة وظائف مخصصة. انقر المجلد Functions في مربع القائمة على اليسار لعرض المجلدات للوظائف المُضمنة ولقاعدة البيانات الحالية. عند تحديد قاعدة البيانات الحالية. يقوم مربع القائمة في المنتصف بعرض الوحدات القياسية في قاعدة البيانات هذه. عند تحديد وحدة قياسية، يقوم مربع القائمة على اليمين بعرض الوظائف المخصصة المخزنة في الوحدة. وعند تحديد وظيفة مُخصصة ونقر الزر Paste فيتم عرض الجملة للوظيفة في مربع التحرير. يوضح الشكل ٣-١٩ الجملة للوظيفة Isloaded في الوحدة القياسية GlobalCode في قاعدة البيانات Epenses.

الشكل ٣-١٩

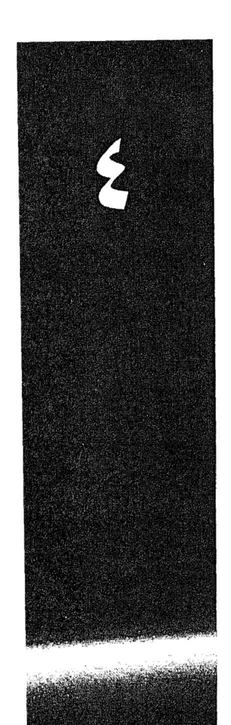
استخدام Expression Builder لعرض الجملة لوظيفة مخصصة.



خلاصة

قمنا من خلال هذا الفصل عمل جولة تقديمية لوحدة كائن أكسس التي تركز على الكائنات والخصائص الشائعة بالنسبة لبرمجة VBA. الأفكار الأساسية هي:

- يحتوي أكسس على كائنان رئيسيان. يكون كل منهما مع تدرج الكائن الخاص به وهما تطبيق أكسس ومحرك قاعدة البيانات.
- عموماً، للإشارة إلى كائن في إجراء VBA يجب عليك استخدام مرجع مؤهل تماماً. للحصول على هذا المرجع، ابدأ من رأس تدرج الكائن وانتقل إلى أسفل الكائن مسجلاً أسماء كل المجموعات والكائنات التي انتقلت خلالها. يمكنك في تدرج الكائن لأكسس استخدام عامل التشغيل النقطة للإشارة إلى الانتقال من كائن لآخر من مجموعاته، واستخدام علامة التعجب على الانتقال من مجموعة إلى أحد أعضائها.
- يمكنك استخدام المجموعات والخصائص الافتراضية لتصغير المراجع.
- للإشارة إلى خاصية كائن. يمكنك تضمين المرجع إلى الكائن واسم الخاصية يفصل بينهما عامل التشغيل النقطة.
- تسمح لك خصائص الكائن Screen الإشارة إلى كائن نشط دون استخدام اسمه.
- يمكنك الإشارة إلى حقل في مصدر السجل الأساسي للنموذج حتى لو لم يكن هناك عنصر تحكم على النموذج المرتبط الحقل.
- يمكنك استخدام Expression Builder لإنشاء المراجع المؤهلة كلية أو غير المؤهلة.



الاتصال بالنماذج

تؤدي النماذج دوراً رئيسياً في معظم تطبيقات أكسس. يعتبر إنشاء تطبيق آلي ليستخدمه الآخرون هو إنشاء واجهة مخصصة تتكون كلية من نماذج وتقارير. يتم إدخال ومراجعة البيانات في النماذج. يتناول هذا الفصل موضوعين أساسيين بالنسبة للنموذج وهما: العلاقة بين النموذج والبيانات التي تم تخزينها في جداول قاعدة البيانات والاتصال بين عناصر التحكم في نموذج أو أكثر.

كما ذكرنا من قبل في هذا الكتاب، تعتبر صفحات الوصول إلى البيانات كائنات تطبيق جديدة مقدمة في أكسس ٢٠٠٠. تتيح صفحات الوصول إلى البيانات للمستخدمين التفاعل مع البيانات في جداول يتم تشغيلها على الإنترنت. تتشابه طريقة أكسس في معالجة صفحات الوصول إلى البيانات مع الطريقة التي يعالج بها النماذج. لذلك يتعلق هذا الفصل بصفحات الوصول إلى البيانات بالإضافة إلى النماذج.

يغطي هذا الفصل العديد من الموضوعات المتقدمة في تصميم النموذج:

- استخدام استعلامات AutoLookup لعرض المعلومات من جدولين أو أكثر والبحث عن معلومات من أحد الجداول آلياً.
- استخدام أوراق بيانات فرعي لعرض وتحرير بيانات مرتبطة أو متصلة في جدول أو استعلام أو نموذج ورقة بيانات أو نموذج فرعي.
- استخدام حقول استعلام محسوبة.
- استخدام عناصر تحكم النموذج المحسوبة للبحث عن المعلومات في عناصر التحكم الأخرى في نفس النموذج أو في نموذج آخر للبحث عن المعلومات في حقول الجداول أو الاستعلام باستخدام دالة تجميع المجال.
- استخدام عناصر تحكم نموذج غير منضمة لضغط القيم المؤقتة.
- استخدام مربعات السرد والتحرير لعرض السجلات من جدول أو استعلام كصفوف لورقة بيانات مصغرة.
- تزامن نموذجين باستخدام تقنية النموذج/النموذج الفرعي المضمنة.
- تزامن نموذجين تم عرضهما في أطر منفصلة من خلال إنشاء إجراءات بواسطة Form Wizard.

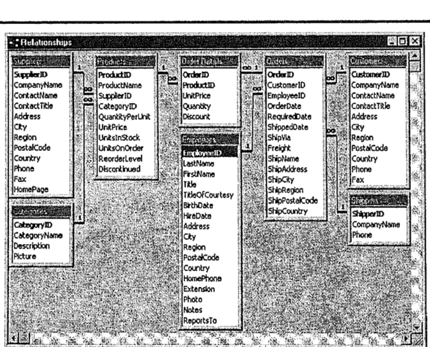
يدور موضوع هذا الفصل عن كيفية انتقال المعلومات من عنصر إلى آخر. كما سنتناول المواقف التي يقوم فيها أكسس بتحديث العرض آلياً وتلك التي ترغب في قيامها بتحديث العرض تفاعلياً أو كتابة البرامج لتحديثه.

ربط النماذج وعناصر التحكم بالبيانات

يعتبر السبب الأساسي وراء استخدام قاعدة البيانات العلائقية لإدارة المعلومات هو تصغير تكرار المعلومات المخزنة من خلال تخزين الحقائق الخاصة بالموضوعات المختلفة في جداول منفصلة. إذا قمت بتصميم قاعدة بيانات علائقية بطريقة صحيحة، ستعتبر الحقول الوحيدة التي من المفترض أن تحتوي على معلومات مكررة هي تلك التي يتم استخدامها لربط الجداول وتظهر البيانات الموجودة في الحقول الأخرى مرة واحدة فقط. إذا تم تغيير البيانات "مثل تغيير عنوان أو إملاء اسم"، يمكنك تنفيذ التغيير في مكان واحد فقط وستظهر هذه البيانات التي وقع عليها التغيير آلياً في الاستعلامات وصفحات الوصول إلى البيانات والنماذج والتقارير.

وعلى العكس تماماً، ففي قاعدة بيانات الملف غير المفصل، قد تظهر نفس البيانات في حقول في سجلات عديدة. فعلى سبيل المثال، قد يكرر كل طلب اسم وعنوان العميل. ففي قاعدة بيانات الملف غير المفصل، لا يعني تغيير البيانات في سجل واحد بدء التغيير الآلي في الحقول الأخرى التي تحتوي على نفس البيانات.

يوضح الشكل ٤-١ الجداول والعلاقات الموجودة في قاعدة بيانات Northwind المرفقة مع أكسس "سيتم استخدام قاعدة البيانات النموذجية هذه في بقية الكتاب". يساعد تخزين البيانات في جداول منفصلة على منح النظام العلائقي طاقته ولكنه يزيد من صعوبته عند الرغبة في مشاهدة واستخدام المعلومات من جداول مختلفة في وقت واحد. يوفر أكسس العديد من الطرق لاسترداد البيانات التي تم تخزينها في جداول متعددة.



الشكل ٤-١

الجداول والعلاقات الموجودة في قاعدة بيانات Northwind.

ملاحظة تستخدم الأمثلة الموجودة في هذا الفصل قاعدة البيانات النموذجية Northwind. لمتابعة ذلك، قم بإنشاء نسخة جديدة من Northwind.mdb واحفظها في مجلد VBAHandbook باسم Northwind_Ch4.mdb. انظر مقدمة الكتاب للتعرف على مزيد من التعليمات الخاصة بإعداد هذا المجلد. افتح Northwind_Ch4.mdb.

ما هي مجموعة السجلات

تعتبر نتيجة فتح جدول أو تشغيل استعلام تم تخزينه أو عبارة SQL هي إنشاء مجموعة سجلات "*recordset*" في الذاكرة بواسطة محرك قاعدة البيانات. تعتبر recordset هي مجموعة السجلات الموجودة في جدول أو تلك الناتجة عن تشغيل استعلام أو عبارة SQL. "لا تنتج جميع الاستعلامات وعبارات SQL سجلات، كما أن استدعاء هذه الاستعلامات الإجرائية لا ينتج السجلات عند تشغيلها".

ملاحظة يتم إرجاع مجموعة سجلات DAO عند طلب مجموعة سجلات في قاعدة البيانات. ويتم إرجاع مجموعة سجلات ADO في مشروع. سنتناول مناقشة مجموعة سجلات ADO وDAO في الفصل ٦.

وفي بعض الأحيان، يتم تخزين البيانات المراد استردادها في جدول فردي. على سبيل المثال، إذا أردت مراجعة عناوين العاملين، تعمل مع جدول Employees. وغالباً ما يتم تخزين البيانات في العديد من الجداول المختلفة. افترض أنك ترغب في مراجعة عناوين العاملين الذين يقوموا بمعالجة الطلبات التي يتم شحنها لأيرلندا. يتم تخزين المعلومات الخاصة بالشحن في جدول Orders لذلك تحتاج للعمل مع الحقول من كل من جدولي Employees وOrders. ويعتبر الدور الأساسي للاستعلام هو ربط البيانات التي يتم تخزينها في جدولين أو أكثر. وسوف نلقي نظرة سريعة عن إنشاء استعلام جديد ومراجعة بعض أساسيات استعلام أكسس الهامة. يمكنك فتح نسخة Northwind_Ch4.mdb واتباع المثال.

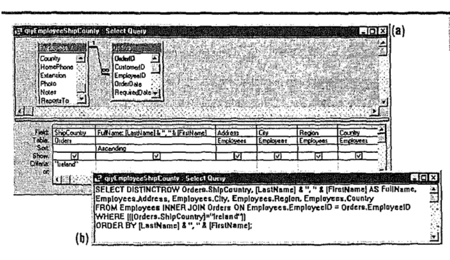
تصميم الاستعلام الرسومي: عند إنشاء استعلام في طريقة عرض تصميم "Design" الاستعلام "انظر الشكل ٤-٢أ"، يتم إضافة الجداول التي تحتوي على البيانات المراد استردادها إلى الجزء العلوي وسحب الحقول من الجداول إلى شبكة التصميم. قم بإضافة جدولي Orders وEmployees. اسحب ShipCountry من جدول Orders وحقول الاسم والعنوان من جدول Employees.

ربط الجداول: يتم الاحتفاظ بعلاقات الجداول التي يتم إنشاؤها في إطار Relationships عند إضافة الجداول إلى استعلام. يمكنك أيضاً إنشاء علاقات استعلام مؤقت جديد بتعديل خطوط الربط للعلاقات الموجودة أو إنشاء خطوط ربط جديدة. يستخدم أكسس خطوط الربط بين الحقول لمطابقة السجلات في الجداول وبناء صف استعلام لكل مجموعة من السجلات المتطابقة. تحتوي جداول Employees وOrdres على علاقة واحد بمتعدد. يشير خط الربط بالأسود العريض والذي يحتوي على رموز 1 و في أي الجانبين إلى تحديد الخيار المراد لتنفيذ التكامل المرجعي في تخطيط Relationships.

إجراء الحسابات: عند إنشاء استعلام، يمكنك تضمين الحقول من الجداول كما يمكنك إنشاء حقول محسوبة جديدة بواسطة إدخال تعبيرات في خلايا Field الفارغة في شبكة تصميم الاستعلام. تعتبر القيمة الموجودة في الحقل المحسوب هي نتيجة التعبير الذي تم إدخاله في خلية Field. لا يتم حفظ القيمة الموجودة في الحقل المحسوب في أي مكان في قاعدة البيانات ويتم إعادة حسابها في كل مرة يتم تشغيل الاستعلام. تعتبر القيمة الموجودة في الحقل المحسوب للقراءة فقط. قم باستبدال حقلي LastName وFirstName بالحقل المحسوب المسمى FullName والذي يؤدي إلى تسلسل البيانات في الحقلين.

تحديد وفرز الصفوف: يمكنك تحديد الصفوف في نتيجة الاستعلام بإدخال التعبيرات في خلايا Criteria وOr في شبكة تصميم الاستعلام. يمكنك فرز الصفوف بإدخال تعليمات الفرز في خلايا Sort. قم بتحديد الصفوف بإدخال Ireland في خلية Criteria أسفل حقل ShipCountry. قم بفرز الحقل المحسوب FullName في ترتيب تصاعدي.

عبارات SQL المتكافئة: عند إنشاء استعلام في طريقة عرض تصميم "Design" الاستعلام، يقوم أكسس آلياً بإنشاء عبارة SQL المتطابقة التي يمكنك ملاحظتها من خلال التغيير إلى طريقة عرض SQL "انظر الشكل ٤-٢ب". يمكنك أيضاً إنشاء الاستعلامات مباشرة كعبارات SQL في طريقة عرض SQL. يمكن إنشاء استعلامات SQL المعينة فقط في طريقة عرض SQL ولا يمكن عرضها في طريقة عرض تصميم "Design" الاستعلام. يعتبر استعلام اتحاد العملاء "Customers" والممولين "Suppliers" في قاعدة بيانات Northwind هو المثال الخاص باستعلام SQL المعين.



الشكل ٤-٢

طريقة عرض Design "أ" وطريقة عرض SQL "ب" لاستعلام qryEmployeesShipCountry.

تخزين الاستعلام: سواء تم إنشاء الاستعلام في طريقة عرض Design أو طريقة عرض SQL، يوجد طريقتان لتخزين الاستعلام وهما:

- يمكنك حفظ الاستعلام باعتباره كائن استعلام في قاعدة البيانات. يتم عرض الاستعلام المحفوظ في لوح Queries لإطار Database. عند حفظ استعلام، يقوم محرك قاعدة البيانات بتحليل الاستعلام وتكوين خطة تقريبية لتشغيله وعند تشغيل استعلام محفوظ، يتم استخدام هذه الخطة المثلى.
- يمكنك تخزين عبارة SQL كإعداد خاصية لكائن آخر. على سبيل المثال، إذا استند نموذج على استعلام، يمكنك تخزين عبارة SQL كخاصية RecordSource للنموذج. في كل مرة يتم فتح النموذج، يقوم محرك قاعدة البيانات بتحليل عبارة SQL وإنشاء خطة تقريبية سريعة وتشغيل الاستعلام.

ملاحظة

لا يمكن استخدام الاستعلامات بواسطة مشروع أكسس. بدلاً من ذلك، تحتمل المشروعات الكائنين الإضافيين لمطور قاعدة البيانات وهما الإجراءات التي تم تخزينها وطرق العرض واللذان يمكن استخدامهما لنفس الأغراض كالاستعلامات. يعتبر الإجراء الذي تم تخزينه هو مجموعة من عبارات SQL التي تم ترجمتها من قبل كما تم تخزينها تحت اسم ما وتشغيلها كوحدة. يمكن تشغيل الإجراء الذي تم تخزينه باستدعائه من التطبيق. تعتبر طرق العرض هي جداول ظاهرية تنتج عن استعلام يتم تخزين التعريف الخاص به في قاعدة البيانات. تساعد تعليمات طرق العرض على تقريب استخدام أكسس مع خادم SQL من خلال تحديد كمية البيانات التي تم تحميلها من الخادم للعناصر التي تهم المستخدم فقط.

يعرض أكسس صفحة البيانات كتصوير مرئي لمجموعة السجلات عند فتح جدول أو تشغيل استعلام تم تخزينه أو عبارة SQL "انظر الشكل ٤-٣". توجد مجموعة السجلات نفسها ككائن موجود في الذاكرة. تختفي مجموعة السجلات كما تتحرر الذاكرة التي تشغلها، عند إغلاق البيانات.

يمكنك فتح جدول أو استعلام مباشرة وعرض بياناته في شبكة طريقة عرض صفحة البيانات "Datasheet view". وعندما يتم فتح نموذج أو تقرير يعتمد على جدول أو استعلام، يرسل أكسس طلب لمحرك قاعدة البيانات لفتح مجموعة سجلات في الذاكرة ثم عرض البيانات من مجموعة السجلات في عناصر تحكم النموذج أو التقرير. عند إغلاق النموذج أو التقرير، يتم إتلاف مجموعة السجلات آلياً. يعالج أكسس الترتيبات باستخدام محرك قاعدة البيانات كما يعالج الاتصال بين البيانات والنموذج أو التقرير. لا تحتاج أثناء العمل إلى معرفة المزيد من التفاصيل عن الاتصال. ومع ذلك، عند تشغيل قاعدة البيانات آلياً، غالباً ما تحتاج إلى كتابة البرامج التي تغير البيانات المعروضة في نموذج أو تقرير، لذلك من المهم فهم اتصال مجموعة السجلات بالنموذج أو اتصال مجموعة السجلات بالتقرير.

qryEmployeeShipCountry : Select Query

Ship Country	FullName	Address	City	Region	Country
Ireland	Callahan, Laura	4726 - 11th Ave. N.E.	Seattle	WA	USA
Ireland	Davolio, Nancy	507 - 20th Ave. E.	Seattle	WA	USA
Ireland	Dodsworth, Anne	7 Houndstooth Rd.	London		UK
Ireland	Dodsworth, Anne	7 Houndstooth Rd.	London		UK
Ireland	Dodsworth, Anne	7 Houndstooth Rd.	London		UK
Ireland	Fuller, Andrew	908 W. Capital Way	Tacoma	WA	USA
Ireland	Fuller, Andrew	908 W. Capital Way	Tacoma	WA	USA
Ireland	Fuller, Andrew	908 W. Capital Way	Tacoma	WA	USA
Ireland	King, Robert	Edgeham Hollow	London		UK
Ireland	King, Robert	Edgeham Hollow	London		UK
Ireland	Leverling, Janet	722 Moss Bay Blvd.	Kirkland	WA	USA
Ireland	Leverling, Janet	722 Moss Bay Blvd.	Kirkland	WA	USA
Ireland	Leverling, Janet	722 Moss Bay Blvd.	Kirkland	WA	USA

Record: 1 of 19

الشكل ٤-٣

تعتبر طريقة عرض صفحة البيانات هي تصوير مرئي على شاشة مجموعة السجلات في الذاكرة.

مصدر السجل لنموذج أو تقرير

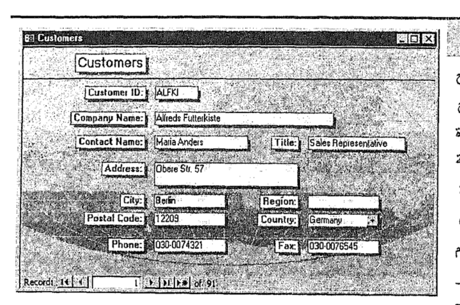
يوجد نوعان من النماذج والتقارير وهما المنضم وغير المنضم. يتم استخدام خاصية RecordSource لتحديد مصدر البيانات للنموذج أو للتقرير عند إنشاء نموذج أو تقرير. يعتبر النموذج أو التقرير غير منضم إذا تركت خاصية RecordSource فارغة، مما يعني أن النموذج أو التقرير لا يحتوي على أي اتصال بمجموعة السجلات.

يمكنك تحديد خاصية RecordSource كاسم جدول أو اسم استعلام محفوظ أو اسم طريقة عرض أو اسم عبارة SQL التي تنتج سجلات في هذه الحالة، يعتبر النموذج أو التقرير منضم *bound* للجدول المحدد أو للاستعلام أو لطريقة العرض أو لعبارة SQL التي تسمى مصدر بيانات النموذج أو التقرير. تعمل عناصر التحكم الموجودة في نموذج أو تقرير منضم مثل الأطر الموجودة في حقول مجموعة السجلات. يقوم محرك قاعدة البيانات بإنشاء مجموعة السجلات المطابقة لإعداد خاصية RecordSource المحدد في ذاكرته عند فتح نموذج أو تقرير منضم.

يوضح الشكل ٤-٤ نموذج Customers استناداً إلى جدول Customer. عند فتح نموذج Customers، يقوم محرك قاعدة البيانات بإنشاء مجموعة سجلات استناداً إلى جدول Customers كما تقوم عناصر التحكم الموجودة في النموذج بعرض البيانات في حقول الصف الأول لمجموعة السجلات. وعند نقر زر التنقل، ينتقل أكسس إلى سجل آخر ويعرض عنصر التحكم البيانات من الصف المطابق لمجموعة السجلات. وعند إغلاق النموذج، تختفي مجموعة السجلات.

ملاحظة

لا تحتوي صفحات الوصول إلى البيانات على خصائص RecordSource ولكنها تتكون من مقاطع هي "مقطع التسمية التوضيحية ورأس المجموعة وتذييل المجموعة ومقطع تنقل السجل" التي تحتوي على عناصر التحكم. تحتوي هذه المقاطع على خاصية RecordSource التي يتم إعدادها لربط المقطع بجدول أو باستعلام أو بطريقة عرض أو بعبارة SQL. وحينئذ يمكنك ربط عناصر التحكم بالمقطع باستخدام خاصية ControlSource "التي سنتناولها في المقطع التالي".



الشكل ٤-٤

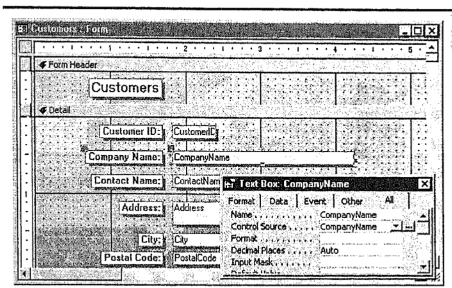
عند فتـــح نمـــوذج Customers، يفتح محـــرك قـــــاعدة البيانات مجموعـــة سجلات في الذاكرة بالاستناد إلى جدول Customers ثـــم تعـــرض عنـــاصر تحكـــم النمـــــوذج البيانات من حقـــول مجموعة السجلات.

مصدر عنصر التحكم لعنصر التحكم

يتم استخدام خاصية ControlSource لتحديد مصدر بيانات عنصر التحكم عند إنشـــاء عنصـــر تحكم بيانات في صفحة الوصول إلى البيانات أو النموذج أو التقرير. يوجد نوعان من عنـــاصر تحكم البيانات التي يمكنك وضعها في صفحة أو نموذج أو تقرير منضم وغير منضم.

عناصر تحكم منضمة

إذا كان مقطع الصفحة أو النموذج أو التقرير منضم لمصدر بيانات، يمكنك حينئذ ربط عنصـــر تحكم في المقطع أو النموذج أو التقرير بحقل في مصدر بيانات المقطع أو النموذج أو التقرير من خلال إدخال اسم الحقل في خاصية ControlSource. يعمل عنصر التحكم المنضم كإطار فـــي الحقـــل المحـــدد. يوضـــح الشـــكل ٤-٥ ربـــط عنصـــر تحكـــم CompanyName بحقـــل CompanyName في مصدر بيانات النموذج.



الشكل ٤-٥ يربط إعداد خاصية ControlSource عنصر التحكم بحقل في مصدر بيانات النموذج.

عناصر التحكم غير المنضمة

يعتبر عنصر تحكم البيانات غير منضم، إذا لم يتم تعيين خاصية ControlSource إلى اسم حقل في مصدر بيانات المقطع أو النموذج أو التقرير. يوجد نوعان من عناصر التحكم غير المنضمة:

- عندما يتم إدخال تعبير في خاصية ControlSource. يسمى عنصر التحكم باسم calculated control. وعندما يتم فتح نموذج أو صفحة أو تقرير باستخدام عنصر تحكم محسوب، يستخدم أكسس التعبير الموجود في خاصية ControlSource آلياً لتكوين قيمة ثم عرضها في عنصر التحكم. يتم تحديد القيمة المعروضة في عنصر التحكم المحسوب بواسطة تعبير ControlSource كما لا يمكنك تغيير القيمة بالكتابة في عنصر التحكم.
- عندما تترك إعداد خاصية ControlSource فارغ، لن يحتوي عنصر التحكم غير المنضم على قيود تحرير. يمكنك إدخال وتحرير القيمة في مثل هذا النوع من عنصر التحكم غير المنضم.

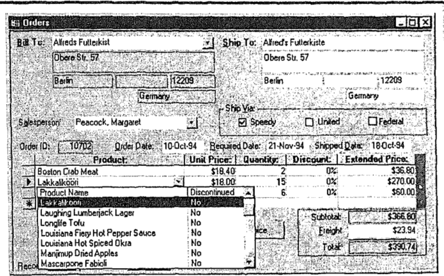
سواء يتم تعيين خاصية ControlSource لتعبير أو تظل فارغة، تتواجد القيمة المعروضة في عنصر التحكم غير المنضم فقط في أثناء فتح النموذج أو الصفحة أو التقرير وتختفي عند إغلاق النموذج أو الصفحة أو التقرير.

عناصر تحكم بمصدري بيانات

يعتبر مربع السرد والتحرير ومربع القائمة عناصر تحكم هامة نظراً لأنه يمكن إرفاقهما بمصدرين مختلفين من البيانات. يحتوي مربع السرد والتحرير أو عنصر تحكم مربع القائمة على خاصية ControlSource التي يمكنك استخدامها لتحديد مصدر بيانات عنصر التحكم. يمكنك ربط مربع سرد وتحرير أو مربع قائمة بحقل موجود في مصدر البيانات الخاص بمقطع صفحة الوصول إلى البيانات أو التقرير أو النموذج أو الإبقاء على عنصر التحكم غير منضم، تماماً كما تترك مربع النص. تحتوي عناصر التحكم هذه على خاصيتي RowSource وRowSourceType التي تستخدمها لتحديد مصدر البيانات المعروض في صفوف القائمة. يمكن أن يكون مصدر البيانات للصفوف الموجودة في القائمة أحد الأنواع الأربعة التالية:

- جدول أو استعلام أو عبارة SQL التي تنتج السجلات.
- قائمة بالقيم التي حددتها.
- قائمة بأسماء الحقول الموجودة في جدول أو استعلام أو عبارة SQL.
- قائمة بالقيم التي حددتها باستخدام الدالة المخصصة.

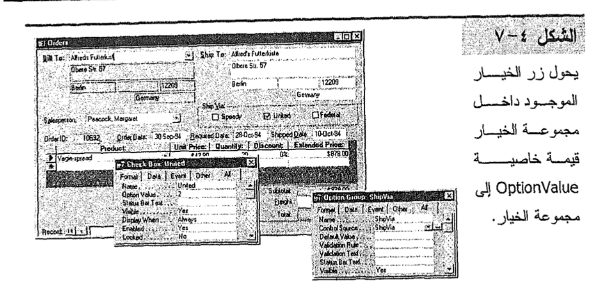
عند فتح نموذج أو صفحة أو تقرير باستخدام مربع قائمة أو قائمة مربع سرد وتحرير منسدلة في نموذج، يقوم أكسس بإنشاء القائمة التي تقوم بتحديدها. إذا كنت تقوم بتحديد جدول أو استعلام أو عبارة SQL، يقوم محرك قاعدة البيانات بإنشاء مجموعة سجلات للقائمة ويعرض أكسس الحقول من مجموعة السجلات الموجودة في الصفوف. مما يعني أنه يمكن إرفاق نموذج أو مقطع صفحة أو تقرير يحتوي على مربع قائمة أو مربع سرد وتحرير بمجموعتين من السجلات وهما: مجموعة سجلات النموذج أو المقطع أو التقرير "المحددة في خاصية "RecordSource ومجموعة سجلات صف عنصر التحكم "المحددة في خاصتي RowSource وRowSource Type لعنصر التحكم". على سبيل المثال، يتم إرفاق النموذج الفرعي بمجموعتي سجلات في نموذج Orders الموضح في الشكل ٤-٦: تستند مجموعة سجلات النموذج الفرعي إلى استعلام Order Details Extended كما تستند مجموعة سجلات صف مربع السرد والتحرير Product إلى جدول Products.



الشكل ٤ – ٦ يحتوي عنصر تحكم مربع السرد والتحرير على طبيعة ثنائية: قد يكون منضم بحقل في مجموعة سجلات النموذج كما تعرض الصفوف من مجموعة سجلات الصفوف.

عناصر تحكم لا تحتوي على خاصية ControlSource

يوجد نوعان من عناصر التحكم لا تحتوي على خاصية ControlSource: عناصر تحكم غير مرفقة ببيانات وأخرى مرفقة ببيانات ولكنها لا تعرض قيمة محددة. على سبيل المثال، عند وضع خانة اختيار أو زر خيار أو مفتاح تبديل مباشرة في النموذج، يحتوي عنصر التحكم على خاصية ControlSource التي يمكنك استخدامها لتحديد مصدر بيانات عنصر التحكم. ومع ذلك، عند وضع أحد هذه العناصر في مجموعة خيار، يصبح عنصر التحكم عنصراً فرعياً "*subcontrol*" ويفقد استقلاله. يتم استبدال خاصية ControlSource لكل عنصر تحكم فرعي بخاصية OptionValue ويحتوي عنصر تحكم مجموعة الخيار فقط على خاصية ControlSource التي يمكنك استخدامها لتحديد مصدر البيانات لمجموعة الخيار. عند تحديد خانة اختيار أو مفتاح تبديل أو زر خيار في مجموعة الخيار، ستساوي قيمة عنصر تحكم مجموعة الخيار OptionValue "قيمة الخيار" لعنصر التحكم الفرعي الذي حددته "انظر الشكل ٤–٧".



الشكل ٤-٧

يحول زر الخيار الموجود داخل مجموعة الخيار قيمة خاصية OptionValue إلى مجموعة الخيار.

يعتبر عنصر تحكم النموذج الفرعي/التقرير الفرعي هو عنصر تحكم آخر لا يحتوي على خاصية ControlSource. في هذه الحالة، لا يعرض عنصر التحكم قيمة ولكن يعرض عنصر تحكم النموذج الفرعي/التقرير الفرعي، بدلاً من ذلك، نموذج أو تقرير آخر. نظراً لأن مصدر عنصر التحكم يعتبر نموذج أو تقرير آخر بدلاً من حقل، يتم استخدام خاصية SourceObject لتحديد اسم النموذج أو التقرير الذي يعرضه عنصر التحكم.

نموذج مصدر السجل الواحد والنموذج الواحد

يستخدم أكسس نموذج مصدر السجل الواحد والنموذج الواحد: حيث يحتوي مقطع الصفحة أو النموذج أو التقرير على مصدر واحد للبيانات. وقد يعرض المقطع أو النموذج أو التقرير البيانات من العديد من الجداول من خلال استخدام استعلام أو عبارة SQL ولكن يمكن إدخال جدول أو استعلام أو طريقة عرض أو عبارة SQL كخاصية RecordSource. عندما يعتمد النموذج على بيانات من جدولين مرتبطين، يمكنك إنشاء استعلام أو طريقة عرض أو عبارة SQL لمصدر السجل. على سبيل المثال، يمكنك إسناد النموذج إلى استعلام qryEmployeesSshipCountry الذي تم إنشاؤه مسبقاً لعرض معلومات الاتصال للعاملين الذين لديهم عملاء في أيرلندا.

قد يبدو النموذج مبنياً على مجموعتين من السجلات بدلاً من مجموعة واحدة استناداً إلى العلاقة بين الجداول وكيفية إنشاء الاستعلام. كمثال، افترض أنك تقوم بإنشاء نموذج لإدخال طلبات جديدة في قاعدة بيانات Northwind. يعرض النموذج معلومات عن عنوان العميل بالإضافة إلى معلومات عن الطلب. يمكنك إنشاء استعلام يعتمد على جدولي Customers وOrders ثم إسناد النموذج إلى الاستعلام. نظراً لأنه يتم التعامل مع عملاء دائمين، يمكن إعداد النموذج بحيث يتم إدخال عناوينهم آلياً. يمكنك تصميم الاستعلام باعتباره استعلام

AutoLookup، لذلك بمجرد أن يتم إدخال CustomerID يبحث الاستعلام عن معلومات العميل ويعرضها آلياً. ثم يمكنك إدخال معلومات عن الطلب في عناصر التحكم المتبقية. يبدو النموذج كأنه على مصدرين للبيانات أحدهما للمعلومات عن العميل والثاني للمعلومات عن الطلب، نظراً لقيام أكسس بملء عناصر تحكم معلومات العميل آلياً وقيام المستخدم بكتابة هذه المعلومات بنفسه في عناصر تحكم إدخال الطلبات.

استعلامات البحث الآلي "AutoLookup"

يعتبر استعلام AutoLookup هو استعلام مبني على جدولين في علاقة واحد بمتعدد. يقوم هذا النوع من الاستعلام بملء قيم الحقول من جدول واحد عند إدخال قيمة في حقل ربط لسجل جديد. يعتبر الاستعلام هو استعلام AutoLookup عندما توفر الشروط التالية:

- يحتوي جدولين في الاستعلام على علاقة واحد بمتعدد. يمكن أن تكون العلاقة مؤقتة تم إنشاؤها في الاستعلام أو تكون علاقة دائمة تم إنشاؤها في إطار Relationship. "ليس بالضرورة فرض التكامل المرجعي".
- يحتوي حقل الربط في أحد جوانب العلاقة على فهرس فريد. مما يعني أنه تم تعيين خاصية Indexed للحقل إلى Yes (No Duplicates). "وليس بالضرورة أن يكون حقل الربط مفتاح أساسي".
- يجب أن يكون حقل الربط الذي تقوم بإضافته إلى الاستعلام من الجانب المتعدد للعلاقة. يعني ذلك بالنسبة لاستعلام AutoLookup المبني على جدولي Customers وOrders، أنه يجب تضمين حقل CustomerID من جدول Orders وليس من جدول Customers.
- يجب أن توجد القيمة التي أدخلتها في حقل ربط صف الاستعلام بالفعل في حقل الربط في جانب واحد. في المثال، يعني ذلك أنه يجب إدخال قيمة CustomerID لعميل موجود مما يعني أنه يمكنك إدخال قيمة CustomerID فقط لسجل تم حفظه بالفعل في جدول Customers.

إذا تم الوفاء بهذه الشروط عند إدخال قيمة في حقل الربط، يبحث أكسس آلياً عن القيم المترابطة من الجدول على الجانب الواحد.

فيما يلي خطوات إنشاء استعلام AutoLookup:

١- قم بإنشاء استعلام جديد مبني على Customers وOrders.

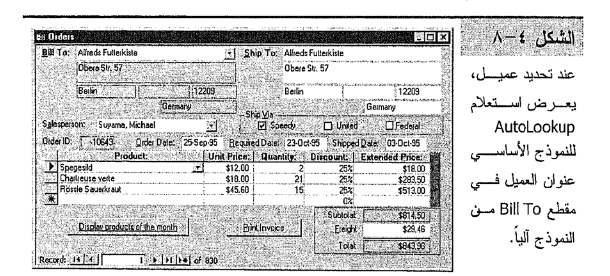
٢- اسحب بعض الحقول التي لا تحتوي CustomerID من Customers إلى الشبكة.

٣- اسحب بعض الحقول من Orders بما في ذلك CustomerID.

٤- قم بالتبديل إلى طريقة عرض Datasheet وحدد عميل من قائمة سرد وتحرير CustomerID. يتم ملء الحقول التي تم سحبها من جدول Customers آلياً.

يستخدم نموذج Orders الموجود في Northwind استعلام AutoLookup كمصدر السجل في النموذج الأساسي. يوضح الشكل ٤-٨ البحث الآلي عن معلومات العميل والذي يحدث عندما يتم إدخال طلب جديد. وعندما يتم تحديد عميل في مربع سرد وتحرير Bill To، يتم إدخال معلومات عن عنوان العميل آلياً في مقطع Bill To من النموذج الأساسي. لاحظ أيضاً إدخال المعلومات آلياً في مقطع Ship To للنموذج، ومع ذلك، لا تعتبر معلومات Ship To هي نتيجة البحث الآلي. وسوف نرجع إلى التقنية المستخدمة لعناصر تحكم Ship To فيما بعد في هذا الفصل.

تعتبر AutoLookup تقنية ذات قيمة وخاصة عند تصميم نموذج لإدخال البيانات في جدول وعند الرغبة في بحث المعلومات آلياً من جدول آخر.



الشكل ٤-٨

عند تحديد عميل، يعرض استعلام AutoLookup للنموذج الأساسي عنوان العميل في مقطع Bill To من النموذج آلياً.

صفحات البيانات الفرعية

تتيح صفحات البيانات الفرعية إمكانية عرض وتحرير البيانات المرتبطة أو المتصلة. يقوم أكسس بإنشاء صفحة بيانات فرعية آلياً، في أي وقت تتوفر علاقة واحد بواحد بالجدول. سوف يقوم أكسس أيضاً بإنشاء صفحات بيانات فرعية في أمثلة تحتوي على علاقة واحد بمتعدد عندما يتم تعيين خاصية Subdatasheet بالجدول إلى Auto. تعتبر Subdatasheet أكثر تقييداً من ناحية التكامل المرجعي حيث يجب أن تحتوي العلاقة على مفتاح أساسي لجدول واحد ومفتاح غريب في جدول آخر. يمكنك إضافة صفحة بيانات إلى أي جدول أو استعلام أو نموذج أو

صفحة بيانات كما يمكنك تحديد الجداول أو الاستعلامات ككائنات المصدر الخاصة بها. قد يتوفر لديك حد أقصى من ثمانية مستويات من صفحات البيانات المضمنة. يمكن أن تحتوي كل صفحة بيانات أو صفحة بيانات فرعية على صفحة بيانات فرعية مضمنة واحدة فقط.

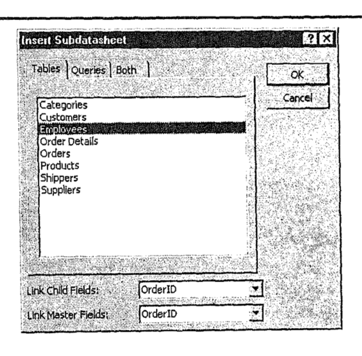
يمكننا إنشاء نموذج صفحة بيانات فرعية توضح البيانات المنضمنة من جدولي Orders وEmployees:

١- افتح قاعدة بيانات Northwind وجدول Orders في طريقة عرض Datasheet.

٢- حدد Insert ⇦ Subdatasheet. يفتح ذلك حوار Insert Subdatasheet. يتيح لك الحوار تحديد نوع الكائن المراد استخدامه كمصدر السجل لصفحة البيانات الفرعية. انقر علامة تبويب Tables "انظر الشكل ٤-٩".

الشكل ٤-٩

حوار Insert Subdatasheet لقاعدة بيانات Northwind.



٣- انقر جدول Employees في مربع القائمة.

٤- يتم استخدام قائمة Link Child Fields المنسدلة للإشارة إلى المفتاح الغريب أو لمطابقة الحقل لصفحة البيانات الفرعية. اختر EmployeeID.

٥- يتم استخدام قائمة Link Master Fields المنسدلة للإشارة إلى المفتاح الأساسي أو لمطابقة الحقل لصفحة البيانات المفتوحة. اختر EmployeeID ثم انقر OK.

٦- انقر + بجوار صف موجود في جدول Orders. يؤدي ذلك إلى توسيع صفحة البيانات الفرعية التي تم إنشاؤها مؤخراً كما يعرض التفاصيل للعاملين الذين قاموا بإدخال الطلب "انظر الشكل ٤-١٠".

Orders : Table

Order ID	Customer	Employee	Order Date	Required Date	Shipped Date
10248	Vins et alcools Chevalier	Buchanan, Steven	04-Jul-1996	01-Aug-1996	16-Jul-1996

Last Name	First Name	Title	Title Of	Birth Date	Hire Date	Address
[illegible]	Steven	Sales Manager	Mr.	04-Mar-1955	17-Oct-1993	14 Garrett Hill

Order ID	Customer	Employee	Order Date	Required Date	Shipped Date
10249	Toms Spezialitäten	Suyama, Michael	05-Jul-1996	16-Aug-1996	10-Jul-1996
10250	Hanari Carnes	Peacock, Margaret	08-Jul-1996	05-Aug-1996	12-Jul-1996
10251	Victuailles en stock	Leverling, Janet	08-Jul-1996	05-Aug-1996	15-Jul-1996
10252	Suprêmes délices	Peacock, Margaret	09-Jul-1996	06-Aug-1996	11-Jul-1996

الشكل ٤-١٠ صفحة البيانات الفرعية Employees لجدول Orders.

ملاحظة لا يمكنك استخدام صفحات البيانات الفرعية في مشروعات أكسس حيث أنها تتاح فقط في قواعد بيانات أكسس.

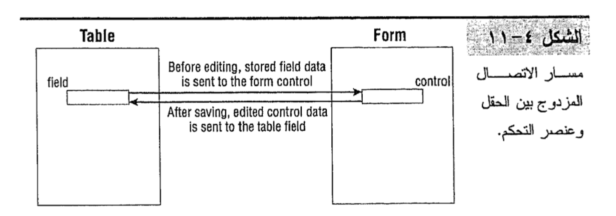
الانتقال بين عناصر التحكم والحقول

عند استخدام استعلام AutoLookup كمصدر سجل للنموذج، يقوم عنصر التحكم المنضم لحقل الربط بدور رائع فعند تغيير القيمة الموجودة فيه، يقوم أكسس آلياً بتغيير القيمة المعروضة في عناصر التحكم المنضمة لحقول البحث. يتم التنقل بين عناصر تحكم الربط وعناصر تحكم البحث. ومع ذلك، عندما لا يستند النموذج إلى استعلام AutoLookup وتنضم جميع عناصر التحكم الموجودة فيه إلى حقول الجدول من مجموعة سجلات النموذج، لا يتوفر الاتصال بين عناصر التحكم. على سبيل المثال، ينضم كل عنصر تحكم في نموذج Employees في Northwind إلى حقل في جدول Employees. لا يتأثر أي عنصر تحكم آخر، إذا تم تغيير القيمة في أي عنصر تحكم، مثل FirstName.

يؤدي ربط عنصر تحكم بحقل جدول إلى إنشاء مسار اتصال مزدوج بين عنصر التحكم والحقل. عندما يتم عرض سجل لأول مرة في نموذج، يقوم كل عنصر تحكم منضم بعرض البيانات التي تم تخزينها في حقل الجدول. عندما يتم إدخال أو تغيير البيانات في عنصر تحكم النموذج، يتم وضع البيانات الجديدة أو المتغيرة في المحزن المؤقت لعنصر التحكم في الذاكرة. يعرض عنصر التحكم القيمة المحررة في المخزن المؤقت ولكن مع استمرار حقل الجدول في تخزين البيانات التي لم يتم تحريرها. عند الانتقال إلى عنصر تحكم آخر لنفس السجل، يقوم أكسس بتحديث المحزن المؤقت لعنصر التحكم واضعاً البيانات المحررة في المحزن المؤقت للسجل في الذاكرة. وفي أثناء تحرير سجل، تعرض عناصر تحكم النموذج التحريرات بينما يستمر الجدول في تخزين السجل الأصلي. يقوم أكسس بتحديث المحزن المؤقت للسجل وحفظ البيانات الجديدة أو المتغيرة إلى سجل الجدول فقط عندما يتم حفظ السجل. تتطابق البيانات المعروضة في عناصر تحكم النموذج والمخزنة في حقول الجدول متماثلة بعد أن يتم حفظ السجل. يرسل حقل الجدول القيمة المخزنة إلى عنصر تحكم النموذج ومن ناحية أخرى، يرسل

عنصر تحكم النموذج القيمة المحررة إلى حقل الجدول عند حفظ السجل. يتم توضيح هذه العملية في الشكل ٤-١١.

عندما ينضم عنصر تحكم إلى حقل جدول، يمكنك إدخال بيانات جديدة أو تغيير البيانات الموجودة بكتابة قيمة في عنصر التحكم. يعتبر كل عنصر تحكم منضم لحقل جدول مستقلاً عن عناصر التحكم الأخرى بالنموذج.



الشكل ٤-١١
مسار الاتصال المزدوج بين الحقل وعنصر التحكم.

يوجد طريقتان يمكنك تنظيمهما لعناصر التحكم الموجودة على النموذج لتبادل المعلومات فيما بينهما: وذلك من خلال استخدام الحقول المحسوبة في مصدر سجل النموذج أو باستخدام عنصر التحكم المحسوب في النموذج.

استخدام حقول استعلام محسوبة

يعتبر الحقل المحسوب "*calculated field*" هو حقل الاستعلام الذي يعرض نتيجة حساب بدلاً من عرض قيمة بيانات مخزنة في جدول قاعدة بيانات. يمكنك إنشاء حقل محسوب بكتابة تعبير في خلية Field الفارغة في شبكة تصميم الاستعلام. تقوم بتسمية الحقل المحسوب بكتابة اسم متبوع بنقطتين على يسار التعبير. "إذا لم تقم بتسمية الحقل، يستخدم أكسس الاسم الافتراضي ExprN حيث يكون N عدد صحيح يقوم أكسس بزيادته لكل حقل محسوب في الاستعلام". يمكن أن يتضمن التعبير عوامل تشغيل ودوال ومراجع إلى حقول أخرى في الاستعلام ومراجع إلى القيم التي يمكن إرجاعها بواسطة كائنات أخرى مثل القيم الموجودة في عناصر التحكم الموجودة في النماذج المفتوحة. على سبيل المثال، افترض أنك ترغب في عرض الاسم الكامل لعامل في عنصر تحكم واحد في نموذج يستند إلى جدول Employees. فيما يلي الخطوات التالية:

١- قم بإنشاء استعلام يستند إلى جدول Employees ويحتوي على كل حقول الجدول في الاستعلام.

٢- أضف حقل استعلام محسوب بإدخال التعبير التالي في خلية Field الجديدة في شبكة تصميم الاستعلام "انظر الشكل ٤-١٢أ":

FullName: LasteName & " , " & FirstName

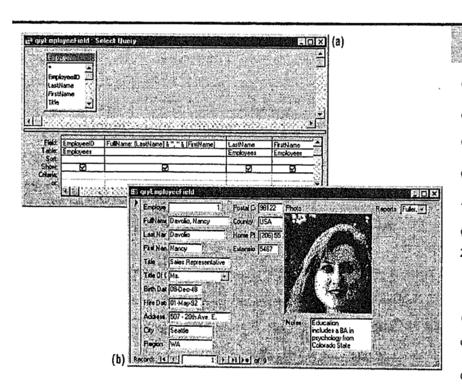
٣- احفظ الاستعلام باعتباره qryEmployeeField.

٤- استخدم AutoForm Wizard لإنشاء نموذج جديد يستند إلى الاستعلام "انظر الشكل ٤-١٢ب".

٥- تأكد من أنه لا يمكنك تغيير القيمة الموجودة في عنصر تحكم FullName.

٦- قم بتغيير القيمة في عنصر تحكم FirstName ثم انتقل إلى عنصر التحكم التالي. عندما تنقر خارج عنصر التحكم المتغير، ينتقل التغيير إلى عنصر تحكم FullName آلياً.

٧- اغلق النموذج واحفظه باعتباره frmEmployeeField.



الشكل ٤-١٢

استخدام حقل محسوب في استعلام "أ" لتسلسل قيمتي حقل. يتم تحديث عنصر التحكم المنضم للحقول المحسوبة آلياً عندما يتم تغيير قيم الحقول "ب" تمّ إعادة ترتيب عناصر التحكم في النموذج في هذا المثال."

استخدام عناصر تحكم النموذج المحسوبة

يمكنك استخدام عنصر تحكم نموذج محسوب كبديل لاستخدام حقل استعلام محسوب. يعتبر عنصر التحكم المحسوب هو عنصر التحكم الذي يعرض نتيجة تعبير. تقوم بإدخال التعبير

مسبوق بعلامة المساواة "=" في خاصية ControlSource لعنصر التحكم. قد يحتوي التعبير على عوامل تشغيل ودوال ومراجع إلى عناصر تحكم أخرى في النموذج ومراجع إلى الحقول في مصدر بيانات النموذج ومراجع إلى القيم التي يمكن إرجاعها بواسطة كائنات أخرى مثل، القيم الموجودة في عناصر التحكم الموجودة في النماذج الأخرى المفتوحة. يتم تعيين خاصية Name لعنصر التحكم إلى اسم مختلف عن اسم أي عنصر تحكم آخر في النموذج.

يمكنك بدلاً من استخدام حقل استعلام محسوب لعرض الاسم الكامل لعامل في عنصر تحكم واحد كما هو موضح في المقطع السابق، استخدام عنصر تحكم محسوب، كما يلي:

١- يؤدي استخدام AutoForm Wizard إلى إنشاء نموذج جديد يستند إلى جدول Employees.

٢- قم بوضع عنصر تحكم غير منضم في النموذج المسمى FullName ثم تعيين خاصية ControlSource إلى هذا التعبير:

= LastName & " , " & irstName

٣- تأكد من أنه لا يمكنك تغيير القيمة في عنصر التحكم المحسوب.

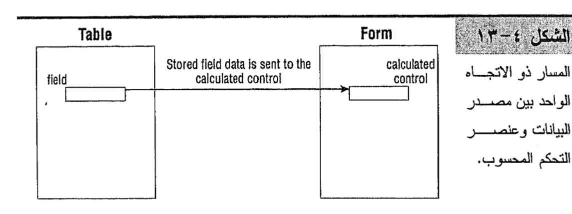
٤- قم بتغيير القيمة الموجودة في عنصر تحكم FirstName ثم الانتقال إلى عنصر التحكم التالي. عند النقر خارج عنصر التحكم المتغير، سيوضح عنصر تحكم FullName القيمة المتغيرة آلياً.

٥- اغلق النموذج ثم احفظه باسم frmEmployeesControl.

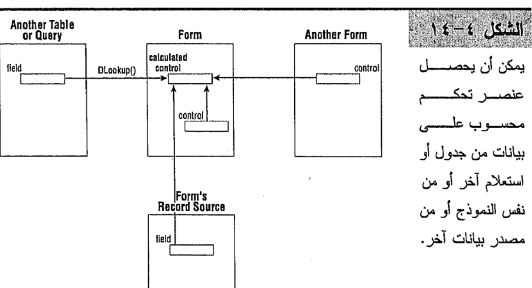
يحتوي عنصر التحكم المحسوب غير المنضم على مسار اتصال واحد إلى البيانات في تعبير ControlSource "انظر الشكل ٤-١٣". يتم تعيين خاصية ControlSource إلى تعبير يسحب البيانات إلى عنصر التحكم. يمكنك استخدام تعبير ControlSource لسحب البيانات من ثلاثة مصادر هم: نفس النموذج أو نموذج آخر مفتوح أو مصدر بيانات آخر باستخدام دالة DLookup() "انظر الشكل ٤-١٤". يعتبر المسار باتجاه واحد نظراً لأن التغييرات التي تحدث في مصدر البيانات تتصل بعنصر التحكم المحسوب ولكن لا يمكن إحداث التغييرات مباشرة لعنصر التحكم المحسوب "لذلك لا يستطيع عنصر التحكم المحسوب إرسال أي تغييرات إلى مصدر البيانات".

سحب البيانات من نفس النموذج

يمكنك سحب البيانات إلى عنصر تحكم غير منضم من عناصر التحكم الأخرى في نفس النموذج أو من الحقول الموجودة في مصدر بيانات النموذج بتعيين خاصية ControlSource إلى تعبير يشير إلى هذه العناصر أو الحقول. يتم استخدام بناء جملة قصير "المرجع غير المؤهل" عندما تشير إلى حقل أو عنصر تحكم. عندما تستخدم بناء الجملة القصير وتشير إلى حقل أو عنصر تحكم باسمه، يفترض أكسس حقل في مصدر بيانات النموذج أو عنصر تحكم في نفس النموذج. يسرد جدول ٤-١ أمثلة التعبيرات لعناصر التحكم المحسوبة التي تسحب البيانات من نفس النموذج.



الشكل ٤-١٣ المسار ذو الاتجاه الواحد بين مصدر البيانات وعنصر التحكم المحسوب.



الشكل ٤-١٤ يمكن أن يحصل عنصر تحكم محسوب على بيانات من جدول أو استعلام آخر أو من نفس النموذج أو من مصدر بيانات آخر.

الجدول ٤-١ : بعض تعبيرات ControlSource لعناصر التحكم المحسوبة التي تحصل على البيانات من نفس النموذج.

تعبير ControlSource	الوصف
=FirstName & " " & LastName	يعرض القيم في عناصر تحكم FirstName و lastName المفصولة بواسطة مسافة.
=Left(CompanyName,4)	يعرض أول أربعة أحرف من قيمة عنصر تحكم CompanyName.
=Sum(Quantity*Price)	يعرض مجموع حاصل ضرب قيم حقلي Quantity و price لجميع السجلات المعروضة بواسطة النموذج.
=Iif(IsNull(Sum(Quantity*Price)) , 0,Sum(Quantity*Price))	يعرض صفر إذا كان المجموع هو Null وبخلاف ذلك يعرض المجموع.
=Count(EmployeeID)	يعرض عدد السجلات المعروضة بواسطة النموذج والتي تحتوي على قيمة non-Null في حقل EmployeeID.
=Count(*)	يعرض عدد السجلات المعروضة بواسطة النموذج. استخدم العلامة النجمية (*) لإحصاء جميع السجلات.

تشير الدالة فقط إلى الحقول الموجودة في مصدر بيانات النموذج، عند استخدام دالة إجمالية مثل Sum() أو Avg() أو Count() في تعبير ما لعنصر تحكم محسوب. يجب استخدام اسم حقل وليس اسم عنصر تحكم للإشارة إلى الحقل، ولا يمكنك الإشارة إلى عنصر تحكم محسوب. إذا أشرت إلى عنصر تحكم منضم لحقل ولكن يحتوي على اسم مختلف عن اسم الحقل، يعرض عنصر التحكم #Name? للإشارة إلى الخطأ. على سبيل المثال، لحساب مجموع القيم الموجودة في عنصر التحكم المسمى GrandTotal والمنضم إلى حقل جدول يسمى Amount، ارجع إلى حقل الجدول باستخدام تعبير =Sum(Amount) بدلاً من تعبير =Sum(GrandTotal).

يمكنك استخدام خصائص كائن Screen في تعبير ControlSource لعنصر التحكم غير المنضم. على سبيل المثال، يمكنك عرض رقم السجل الحالي في عنصر تحكم غير منضم من

خلال إعداد خاصية ControlSource إلى Screen. ActiveForm.CurrentRecord=. عند الانتقال إلى سجل مختلف، سيقوم أكسس آلياً بإعادة حساب القيم الجديدة في عناصر التحكم غير المنضمة في النموذج. قم باستكشاف كائن Screen كما يلي:

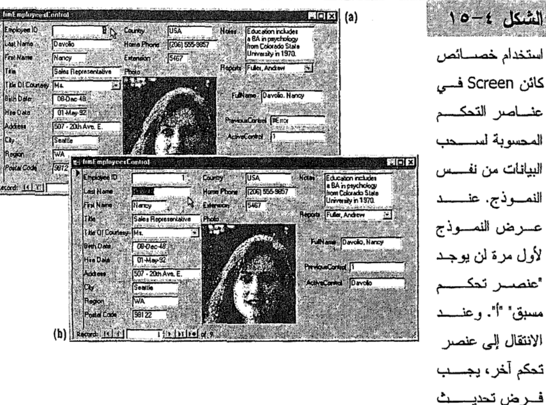
١- قم بوضع مربعي نص غير منضمين في نموذج FrmEmployeesControl. قم بتسمية أحد العناصر باسم txtPrevious وزوده بخاصية ControlSource وهي Screen.PreviousControl= وقم بتسمية عنصر التحكم الآخر وهو txtActive وزوده بخاصية ControlSource وهي Screen.PreviousControl=.

٢- عندما يتم التبديل إلى طريقة عرض Form، يعرض عنصر تحكم txtActive قيمة Employee ID وهي "1" في عنصر التحكم النشط. يقوم عنصر تحكم txtPrevious بعرض Error# حيث أنه لم يكن هناك عنصر تحكم يحتوي على التركيز مسبقاً عندما تم فتح النموذج لأول مرة "انظر الشكل ٤-١٥أ".

٣- انتقل إلى عنصر التحكم التالي بدون القيام بأي تغييرات. لن يتم تحديث txtActive أو txtPrevious آلياً.

٤- افرض تحديث عناصر التحكم غير المنضمة بتحديد Records ⇦ Refresh أو بضغط المفتاح الوظيفي F9 "انظر الشكل ٤-١٥ب".

٥- حرر عنصر تحكم LastName ثم انتقل إلى عنصر التحكم التالي. يتم تحديث عناصر التحكم المحسوبة FullName وtxtPreviuos آلياً.



الشكل ٤-١٥

استخدام خصائص كائن Screen في عناصر التحكم المحسوبة لسحب البيانات من نفس النموذج. عند عرض النموذج لأول مرة لن يوجد "عنصر تحكم مسبق" "أ". وعند الانتقال إلى عنصر تحكم آخر، يجب فرض تحديث عناصر التحكم المحسوبة "ب".

عرض البيانات الحالية

في هذا المثال، لا يتم تحديث عناصر التحكم غير المنضمة آليا إلا إذا قمت بتحرير عنصر تحكم في النموذج. يمكنك فرض تحديث عناصر التحكم المحسوبة بضغط المفتاح الوظيفي F9. لابد من البرمجة للتوصل إلى سلوك تحديث آلي ثابت. يمكنك كتابة برنامج لتحديث السجل عند الانتقال إلى عنصر تحكم ثاني سواء تم تحرير عنصر التحكم الأول أو يتم تحريره.

تعتبر الفكرة وراء إنشاء برنامج تحديث هي تقرير الحدث الذي يجب أن يقوم ببدء تشغيل البرنامج. عند الانتقال إلى عنصر تحكم مختلف في نفس النموذج، سيتعرف عنصر التحكم الذي قمت بنقله على حدثي Exit و lostFocus. كما سيتعرف عنصر التحكم الذي انتقلت إليه على حدثي Enter و gotFocus. يمكنك استخدام أي من هذه الأحداث لبدء تشغيل البرنامج كلما تنقلت بين عناصر التحكم الموجودة في نفس النموذج. سيتطلب اختيار الحدث مهارة أكثر إذا تم التنقل إلى الوراء وإلى الأمام بين

عناصر التحكم الموجودة في نموذجين مختلفين أو في نموذج أساسي ونموذج فرعي حيث قد لا تتعرف جميع عناصر التحكم على كل هذه الأحداث. "انظر الفصل ٢ للتعرف على مزيد من المعلومات عن وقت تشغيل هذه الأحداث".

قد لا تكون البيانات المعروضة في الإطار النشط حديثة لعدة أسباب: وفيما يلي بعض الأمثلة:

- عندما يتوفر لديك إطارين مفتوحين أو عندما تقوم بعرض البيانات في بيئة متعددة المستخدمين، قد لا تظهر التغييرات الموجودة في إطار واحد آلياً في الإطار الآخر.
- قد لا يتم إعادة حساب القيم الموجودة في عناصر التحكم المحسوبة في نموذج ما آلياً حتى تنتقل إلى سجل مختلف.
- لا يتم تحديث البيانات الموجودة في مربع القائمة أو مربع التحرير والسرد لحقل البحث من جدول آخر آلياً عند إحداث تغييرات بالجدول الآخر.

يوجد ثلاثة طرق لتحديث العرض

- قم بتحديث البيانات الحديثة في السجل الموجود. لتحديث الإطار النشط، اختر Records ⇦ Refresh أو اضغط F9. لا يؤدي التحديث إلى تسجيل السجلات أو عرض السجلات المضافة أو إزالة السجلات المحذوفة "يشار إلى السجل المحذوف بالقيمة Deleted # في كل حقل من السجل" أو إزالة السجلات التي لم تعد تفي بمعايير التصفية أو الاستعلام المحدد. يجب إعادة استعلام النموذج لهذه التحديثات.
- يمكنك إعادة حساب عناصر التحكم المحسوبة بضغط F9.
- تؤدي إعادة الاستعلام إلى إعادة تشغيل الاستعلام الذي يستند إليه الكائن أو إذا استند الكائن إلى جدول، سيؤدي إعادة الاستعلام إلى إعادة تشغيل الجدول وعرض سجلات الجدول الحالي فقط. يمكنك إعادة استعلام النموذج بفاعلية بضغط Shift+F9.

يمكنك تنفيذ آلية التحديث وإعادة الاستعلام وإعادة حساب العمليات باستخدام إجراءات VBA.

لاحظ أنه إذا احتوى النموذج على مربع تحرير وسرد أو مربع قائمة يعرض الصفوف من مصدر بيانات غير مصدر بيانات النموذج، لن يقوم إعادة استعلام مصدر بيانات النموذج بإعادة استعلام مصدر بيانات مربع التحرير والسرد أو عنصر تحكم مربع القائمة. يمكنك ملاحظة هذا السلوك مع نماذج Products و categories في Northwind. يعتبر مصدر بيانات نموذج Products هو جدول Products ومصدر بيانات مربع التحرير والسرد Category للنموذج هو عبارة عن عبارة SQL التي

تستند إلى جدول Categories. قم بفتح النموذجين في طريقة عرض Form وأضف فئة جديدة في نموذج Categories واحفظ السجل الجديد. عند عرض قائمة تحرير وسرد Category في Products، لن يتم عرض الفئة الجديدة. يؤدي ضغط Shift+F9 إلى إعادة استعلام مصدر بيانات النموذج ولكنه لا يؤدي إلى إعادة استعلام مصدر بيانات مربع التحرير والسرد. يؤدي ضغط F9 إلى تحديث النموذج وإعادة استعلام مربع التحرير والسرد ولكنه لا يؤدي إلى إعادة استعلام النموذج. تتكون عملية تحديث نموذج Products بفاعلية من خطوتين هما: ضغط F9 ثم ضغط Shift+F9. تعتبر الطريقة الأخرى لإعادة استعلام كل من النموذج وعناصر تحكم مربع التحرير والسرد ومربع القائمة هي إغلاق وإعادة فتح النموذج.

سحب البيانات من نموذج آخر مفتوح أو من تقرير مفتوح

يمكنك سحب البيانات إلى عنصر تحكم غير منضم من عناصر التحكم الموجودة في نموذج مفتوح أو من الحقول الموجودة في مصدر بيانات نموذج آخر بإعداد خاصية ControlSource لعنصر التحكم إلى تعبير يشير إلى تلك عناصر التحكم أو الحقول. في هذه الحالة، يتم استخدام المرجع المؤهل تماماً نظراً للرجوع إلى عناصر التحكم أو الحقول الموجودة في نموذج آخر مفتوح. على سبيل المثال، باستخدام كل من نموذجي Orders وemployees مفتوحين في Northwind، يمكنك عرض الأسماء الكاملة للعاملين قيم نموذج Orders بوضع عنصر تحكم غير منضم في نموذج Orders وبإعداد خاصية ControlSource كما يلي:

```
=Form!Employee!LastName & " , " & Forms!Employee!FirstName
```

يسرد جدول ٤-٢ أمثلة تعبيرات ControlSource التي تسحب البيانات من نموذج آخر مفتوح. في هذه الأمثلة، يوجد عنصر التحكم غير المنضم في نموذج Orders ونموذجه الفرعي كما تشير عناصر التحكم الموجودة في نموذج Customer Orders في Northwind.

الجدول ٤-٢: بعض تعبيرات ControlSource للإشارة إلى نموذج آخر مفتوح

تعبير ControlSource	الوصف
=Forms!{Customer Order}!{Customer Orders Suborm1}!OrderData	يعرض قيمة حقل CustomerID في مصدر بيانات نموذج Customer Orders.

الجدول ٤-٢: بعض تعبيرات ControlSource للإشارة إلى نموذج آخر مفتوح

تعبير ControlSource	الوصف
={Orders Suborm}!UnitPrice	يعرض قيمة عنصر تحكم UnitPrice للسجل الحالي في Orders Subform للنموذج النشط. يتم تعيين موقع عنصر التحكم المحسوب في النموذج الأساسي.
=Parent!OrderID	يعرض قيمة عنصر تحكم OrderID في النموذج الأصل للنموذج الفرعي الحالي. يتم تعيين موقع عنصر التحكم المحسوب في النموذج الفرعي.

يمكنك ربط المراجع بنموذجين مفتوحين في تعبير، على سبيل المثال، بفتح نموذجي Customers وEmployees قم بما يلي:

١- ضع عنصر تحكم غير منضم مسمى txtBoth في نموذج Customers الذي يعرض عدد السجلات الحالية للنموذجين المفتوحين بتعيين خاصية ControlSource كما يلي:

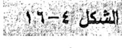
```
="The customers current record is " & Screen.ActiveForm. _
CurrentRecord & " and the Employee current record is " & _
Form!Employee.CurrentRecord
```

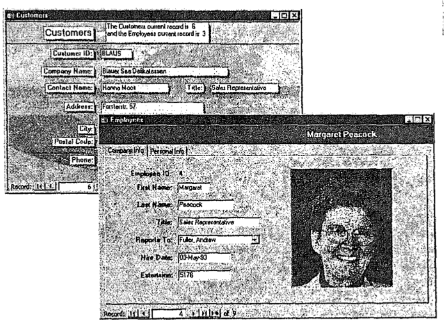
٢- استعراض السجلات الموجودة في نموذج Customer مع ملاحظة تحديث التحكم آلياً لعرض رقم السجل للنموذج الحالي "انظر الشكل ٤-١٦أ".

٣- استعراض السجلات في نموذج Employees ثم نقر نموذج Customers مع ملاحظة عدم تحديث عنصر تحكم txtBoth آلياً "انظر ٤-١٦ب".

٤- فرض إعادة حساب عناصر التحكم في نموذج Customers بالانتقال إلى سجل آخر أو باختيار Records ⇦ Refresh أو بضغط F9.

يمكنك تنفيذ التحديث آلياً من خلال البرمجة. فيمكنك إنشاء إجراء VBA لتحديث نموذج Customers عند نقره حيث يتعرف النموذج على حدث Activate، وبالتالي يمكنك استخدام حدث Activate لبدء تشغيل البرنامج.





الشكل ٤-١٦ يتم تحديث عنصر التحكم المحسوب آلياً عند استعراض سجلات نموذج "أ" Customers وليس سجلات نموذج "ب" Employees.

استخدام ()Dlookup والدوال التكاملية الأخرى

في بعض الأحيان، ترغب في قيام النموذج ببحث المعلومات التي يتم تخزينها في جدول أو استعلام لا يعتبر مصدر السجل الضمني للنموذج. وفي هذه الحالة، يمكنك استخدام دالة ()Dlookup في خاصية ControlSource لعنصر تحكم محسوب لعرض البيانات من جدول أو استعلام آخر. على سبيل المثال، عند إدخال منتج جديد في نموذج Products، قد ترغب في قيام النموذج بحث وعرض وصف الفئة من جدول Categories. يمكنك إنشاء عنصر تحكم غير منضم في نموذج Products وتعيين خاصية ControlSource لعنصر التحكم إلى دالة ()Dlookup تبحث الوصف مباشرة من جدول Categories. يمكنك استخدام دالة ()Dlookup كما يلي:

في استعلام: في تعبير حقل محسوب في خلية Field لتحديد المعايير في خلية Criteria أو في تعبير موجود في خلية Update To في استعلام يتم تحديثه.

في إجراء VBA: في وسيطة شرط أو طريقة.

في نموذج أو تقرير: في عنصر تحكم محسوب.

تحتوي دالة ()Dlookup على ثلاثة وسائط. في أبسط الحالات، تعتبر الوسيطة الأولى هي اسم الحقل الموجود في الجدول أو الاستعلام الذي يحتوي على البيانات المراد بحثها أما الوسيطة الثانية فتعتبر اسم الجدول أو الاستعلام "المجال" والثالثة هي شرط البحث الذي يتم استخدامه لتحديد السجل. يعتبر بناء الجملة هو:

DLookup(fieldname", "tablename" or "queryname",
"searchcondition")

يجب التعبير عن جميع الثلاث وسائط كسلاسل. إذا قام شرط البحث بإرجاع أكثر من شـرط، تقوم دالة Dlookup بإرجاع قيمة الحقل في السجل الذي يفي بالشروط. إذا لم يتم تحديد شـرط بحث، تقوم دالة Dlookup بإرجاع قيمة الحقل من سجل عشوائي في المجال.

للتعرف على كيفية عمل دالة Dlookup():

١- ضع مربع نص غير منضم مسمى txtDescription في نموذج Products وقم بتعيين خاصية ControlSource إلى التعبير التالي:

=DLookup("Description","Categories","CategoryID = Forms! _
Products!CategoryID")

يخبر شرط البحث الموجود في هذا التعبير أكسس بتحديـد السـجل الـذي يطـابق حقـل CategoryID الخاص به القيمة الموجودة في عنصر تحكم CategoryID للنموذج. يعتبر بنـاء الجملة لشرط البحث هو:

fildname = Form!ormname!controlname

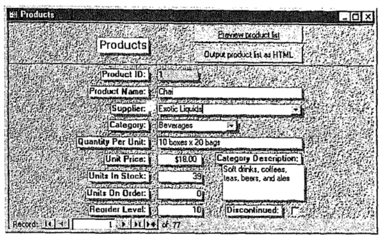
حيث يعتبر الجانب الأيسر لأمر لشرط البحث هو اسم الحقـل الموجـود فـي الجـدول أو الاستعلام الذي يتم بحثه ويعتبر الجانب الأيمن هو المرجع المؤهل تماماً لعنصر تحكم النمـوذج مع القيمة التي يتم بحثها.

٢- استعرض السجلات الموجودة في نموذج Products. لاحـظ تحديـث مربـع نـص Category Description آلياً "انظر الشكل ٤-١٧".

٣- حدد فئة مختلفة من مربع تحرير وسرد Category. لاحظ عدم تحديث عنصر التحكـم غير المنضم آلياً.

٤- قم بتحديث عنصر التحكم بفاعلية بضغط F9.

تعتبر دالة Dlookup() هي مثال لدالة تجميع المجال وهي دالة أكسس المضمنة التي يمكنـك استخدامها لإجراء الحسابات التي تستند إلى القيم الموجودة في حقل جدول أو اسـتعلام. يمكنـك تحديد معايير لتحديد مجموعة السجلات الموجودة في جدول أو استعلام ترغب فـي اسـتخدامه لإجراء الحسابات. تعتبر معايير التحديد اختيارية، فإذا لم تقم بتحديـد معـايير إضافيـة، سـيتم استخدام جميع السجلات الموجودة في الجدول أو الاستعلام. يطلق على الجدول أو الاستعلام اسم مجال *"domain"*. يجب أيضاً تحديد الحقل المراد للدالة العمل معه فبدلاً من تحديد حقل، يمكنك



الشكل ٤-١٧ يستخدم عنصر تحكم Category Description دالة DLookup() لبحث الوصف الموجود في جدول Categories.

تحديد تعبير يقوم بإجراء الحساب على القيم الموجودة في حقل". بمجرد أن يتم تعيين المجال والمعايير وتحديد حقل معين، سوف تقوم الدالة بإجراء حساب على القيم الموجودة في الحقل وإرجاع نتيجة الحساب. يتم سرد دوال تجميع المجال في الجدول ٤-٣.

الجدول ٤-٣: دوال تجميع المجال

الدالة	الوصف
DLookup()	تقوم بإرجاع القيمة الموجودة في الحقل المحدد.
DMin(), DMax()	تقوم بإرجاع الحد الأدنى أو الأقصى للقيمة في الحقل المحدد.
DFirst(), DLast()	تقوم بإرجاع القيمة في الحقل المحدد من أول أو آخر سجل حقيقي.
DAvg()	تقوم بإرجاع المتوسط الحسابي للقيم في الحقل المحدد.
DSum()	تقوم بإرجاع إجمالي القيم في الحقل المحدد.
DStDev(), DstDevP()	تقوم بإرجاع الانحراف القياسي أو الانحراف القياسي للسكان للحقل المحدد.
DVar(), DvarP()	يقوم بإرجاع التباين أو تباين السكان للحقل المحدد.
Dcount()	يقوم بإرجاع عدد السجلات التي تحتوي على قيم غير ملغاة في الحقل المحدد.

يمكنك استخدام أي من دوال تجميع المجال في تعبير ControlSource. تستخدم جميع دوال تجميع المجال نفس بناء الجملة. وفي أكثر الحالات شيوعاً، قد تكون الوسيطة الأولى إما اسم حقل لأحد الحقول الموجودة في المجال أو تعبير يستند على الأقل إلى أحد الحقول. تعتبر الوسيطة الثانية هي مجموعة السجلات الموجودة في الجدول أو الاستعلام "المجال". أما بالنسبة إلى للوسيطة الثالثة، فهي شرط البحث الذي يحدد المجموعة إلى مجموعة أخرى اصغر "المجال المحدد". تعتبر الوسيطة الثالثة اختيارية، فإذا لم تحدد شرط بحث، تستخدم الدالة المجموعة الأكبر من السجلات "المجال". إذا لم يفي أي سجل بشرط البحث أو إذا لم يشتمل المجال على أي سجل، تقوم دالة تجميع المجال بإرجاع قيمة خالية. يعتبر بناء الجملة هو:

```
DFunction("fieldname" or "expression", "tablename" or "queryname",
                                               _"searchcondition")
```

عندما لا يتوفر شرط بحث للحد من المجال، يعتبر بناء الجملة هو:

```
DFunction("fieldname" or "expression", "tablename" or "queryname")
```

على سبيل المثال، يمكنك استخدام دالة Dsum() لإرجاع إجمالي مجموعة من القيم. يعوض تعبير ControlSource: إجمالي قيم حقل Freight في جدول Orders لجميع طلبات العميل المعروضة حالياً في نموذج Customers. يعرض تعبير ControlSource:

```
=DSum("Freight", "Orders",
"CustomerID=Forms!Customers!CustomerID")
```

إجمالي حاصل ضرب حقلي Quantity وUnitPrice في جدول Order Details للمنتج المعروض حالياً في نموذج Products. يوجد عنصر التحكم المحسوب في نموذج Products.

ملاحظة تعتبر دوال تجميع المجال هي عبارات SQL في تنسيق مختلف. يمكنك التفكير في دالة تجميع مجال باعتبارها استعلام يقوم بإرجاع قيمة فردية. يعتبر شرط البحث مساو لعبارة SQL WHERE بدون كلمة WHERE. في كل مرة يتم استخدام أحد هذه الدوال، يتم تشغيل استعلام كعبارة SQL. قبل تشغيل عبارة SQL، يجب أن يقوم محرك قاعدة البيانات بتحليل العبارة لتحديد الطريقة المثلى لتنفيذها. وبالتالي، قد تكون دوال تجميع المجال أبطأ من البدائل الأخرى للبحث عن المعلومات.

توجد أيضاً دوال تجميع SQL، التي يتم تشغيلها بنفس طريقة دوال تجميع المجال. ومع ذلك، بخلاف دوال تجميع المجال، لا يمكن استدعاء دوال SQL مباشرة من الفيجوال بيسك. نظراً لاستخدام الدوال لعبارات SQL بدلاً من الفيجوال بيسك، تعتبر دوال تجميع SQL أكثر فاعلية من دوال تجميع المجال. يتم سرد دوال تجميع SQL في جدول ٤-٤.

الجدول ٤-٤: دوال تجميع SQL

الدالة	الوصف
Avg	تقوم بإرجاع متوسط مجموعة من القيم في حقل أو استعلام.
Count	تقوم بإرجاع عدد من السجلات التي تم إرجاعها بواسطة استعلام.
First, Last	تقوم بإرجاع قيمة حقل من أول أو أخر سجل في نتائج استعلام.
Min, Max	تقوم بإرجاع الحد الأدنى أو الأقصى لمجموعة قيم حقل أو استعلام.
StDev, StDevP	تقوم بإرجاع تقييم الانحراف القياسي من نموذج.
Sum	تقوم بإرجاع إجمالي المجموعة.
Var, VarP	تقوم بإرجاع تقييم التباين "مربع الانحراف القياسي".

استخدام عنصر تحكم غير منضم كمتغير

يعتبر المتغير "variable" في مصطلحات البرمجة هو موقع تخزين مؤقت باسم في الذاكرة التي يتم استخدامها للاحتفاظ بقيمة. يمكنك التفكير في عنصر تحكم نموذج غير منضم كنوع متغير. لا يتم تخزين القيمة المعروضة في عنصر تحكم غير منضم في أي مكان في قاعدة البيانات وتوجد فقط كلما كان النموذج مفتوحاً مما يعني أن مدة متغير عنصر التحكم غير المنضم هو الفاصل الزمني عند فتح النموذج. يعرف عنصر التحكم غير المنضم بأنه متغير عمومي نظراً لإتاحة قيمته لكل الاستعلامات الأخرى والنماذج والتقارير الموجودة في قاعدة البيانات. "ستتعرف في الفصول القادمة على طرق إنشاء المتغيرات بدون استخدام عناصر تحكم النموذج كحاويات تخزين مؤقت".

سحب ودفع البيانات في عنصر تحكم غير منضم

يوجد طريقتان لتعيين قيمة في عنصر تحكم غير منضم. يمكنك استخدام خاصية CcontrolSource لعنصر التحكم غير المنضم لسحب البيانات إلى عنصر التحكم. كما يمكنك سحب البيانات من عناصر تحكم نماذج أو تقارير أخرى ومن حقول موجودة في مصدر السجل الضمني للنموذج ومن الحقول الموجودة في جدول أو استعلام آخر "باستخدام Dlookup() أو أي دالة تجميع مجال أخرى".

يمكنك أيضاً وضع عنصر تحكم غير منضم مع ControlSource فارغة في نموذج. يجب تنفيذ إجراء معين لتعيين قيمة في مثل عنصر التحكم هذا. سيؤدي هذا الإجراء إلى دفع البيانات في عنصر التحكم غير المنضم. يمكنك دفع قيمة في عنصر تحكم غير منضم مع خاصية ControlSource بكتابة القيمة مباشرة من لوحة المفاتيح أو دفعها باستخدام إجراء VBA. يتم توضيح هذه الطرق في الشكل ٤-١٨. يتم توضيح هذه الطرق في الشكل ٤-١٨.

الشكل ٤-١٨

دفع البيانات في عنصر تحكم غير منضم.

على سبيل المثال، يعتبر مربع تحرير وسرد البحث الذي تم وضعه في Expense Reports بواسطة نموذج Employee في الفصل ١ هو عنصر تحكم غير منضم يحتوي على خاصية Control Source فارغة. يحتوي هذا المتغير على قيمة EmployeeID للعميل المراد بحثه. يؤدي اختيار قيمة من مربع التحرير والسرد إلى دفعها في عنصر التحكم. بعد اختيار القيمة، يؤدي إجراء VBA إلى مزامنة النموذج مع القيمة الموجودة في متغير مربع التحرير والسرد.

استخدام نموذج متغيرات عمومية

استناداً إلى ما تحاول القيام به، يمكنك وضع عناصر التحكم غير المنضمة كمتغيرات مباشرة في نماذج المهام في التطبيق أو يمكنك إنشاء نموذج متغيرات عمومي منفصل ليحتوي على بعض المتغيرات العمومية. إذا أردت إتاحة بعض القيم في المرة التالية لتشغيل التطبيق، قم بإنشاء جدول لتخزين القيم المراد إبقائها ثابتة أو دائمة. عموماً قد يحتوي نموذج المتغيرات العمومي

على كل من عناصر تحكم غير منضمة للمتغيرات الانتقالية وأخرى منضمة للمتغيرات الدائمة. يجب أن يتم فتح النموذج متى تم فتح التطبيق حتى تتاح المتغيرات للنماذج والاستعلامات الأخرى. يجب أن يكون النموذج مرئياً للمستخدم إذا أراد كتابة القيم مباشرة في بعض عناصر التحكم. ومع ذلك، يمكن إخفاء النموذج إذا تم سحب القيم في عناصر التحكم المحسوبة أو في تعبيرات الاستعلام أو تم سحبها إلى عناصر التحكم التي تحتوي على إجراءات VBA.

حفظ نتيجة محسوبة لقاعدة البيانات

في بعض الأحيان، تقوم باستخدام تعبير ما لإجراء حساب في نموذج وترغب في حفظ النتيجة إلى قاعدة البيانات. على سبيل المثال، افترض أنك تقوم بحساب الدخل الإجمالي للطلبات الخاصة بكل عميل وترغب في حفظ المجموع في جدول Customers.

ملاحظة تعتبر أحد إرشادات تصميم قاعدة البيانات هي أنك لا تقوم بتخزين القيم المحسوبة ولكن هي أنك تقوم بإعادة حسابها في كل مرة تحتاج إليها. ومع ذلك، يعتبر من الأفضل والأسرع بالنسبة لقيم التلخيص الخضوع لهذه القاعدة وتخزين القيم.

عندما يتم استخدام عنصر تحكم محسوب للحساب، تقوم باستخدام خاصية ControlSource للاحتفاظ بالتعبير لذلك لا يمكنك استخدام خاصية ControlSource أيضاً لربط عنصر التحكم بحقل الجدول. يعتبر حل هذه المشكلة هو تنفيذ الحساب في إجراء VBA بدلاً من استخدام خاصية ControlSource ثم وضع نتيجة الحساب في عنصر التحكم المنضم باستخدام إجراء VBA آخر. وسوف نتابع كيفية وضع القيم في عناصر التحكم المنضمة وغير المنضمة في الفصول اللاحقة.

استخدام مربع تحرير وسرد أو مربع قائمة لبحث المعلومات

في أكسس ٢٠٠٠، يحتوي النموذج على جدول واستعلام وطريقة عرض واحدة فقط أو على عبارة SQL كمصدر سجله. كما تم التوضيح في المقاطع السابقة، يمكنك تجنب تحديد مصدر بيانات واحد لكل نموذج من خلال بحث القيم الموجودة في نموذج أو تقرير آخر مفتوح أو باستخدام دالة ()Dlookup لبحث القيم الموجودة في أي جدول أو استعلام. ينتج عن هذه التقنيات قيمة واحدة يمكنك سحبها إلى عنصر تحكم غير منضم باستخدام خاصية ControlSource أو دفعها في عنصر تحكم منضم أو آخر غير منضم يحتوي على خاصية ControlSource فارغة. يمكن الاستفادة من هذه التقنيات عندما ترغب في عرض قيمة واحدة تستند إلى مصدر بيانات

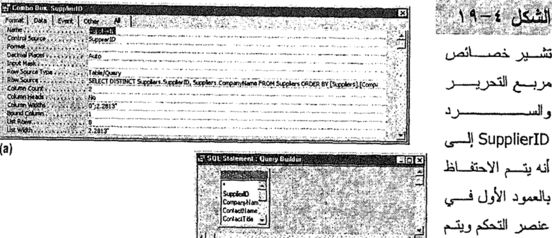
آخر ولا يمكن الاستفادة منها عندما ترغب في عرض قيم متعددة في نفس الوقت. تتيح عناصر تحكم مربع التحرير والسرد ومربع القائمة طريقة فعالة لبحث وعرض صفوف كاملة من القيم من مصدر بيانات ثاني.

يحتوي مربع التحرير والسرد أو مربع القائمة على خاصيتي RowSource وRowSource Type اللتان يمكنك استخدامهما لتحديد مصدر بيانات لعنصر التحكم الذي قد يختلف عن مصدر بيانات النموذج. مما يعني أنه يمكنك عرض السجلات من الجدول أو الاستعلام أو طريقة العرض أو عبارة SQL في خاصية RowSource كصفوف صفحة بيانات مصغرة "mini-datasheet". تسحب خاصية RowSource صفوف البيانات بأكملها من مصدر بيانات إلى قائمة مربع التحرير والسرد. عند عرض مربع التحرير والسرد، يتم عرض صفحة البيانات المصغرة لمصدر البيانات الثاني. يحتوي مربع التحرير والسرد أيضاً على خاصية ControlSource وخاصية BoundColumn. يمكنك استخدام خاصية BoundColumn لمربع التحرير والسرد أو مربع القائمة لتحديد العمود المراد استخدامه في صفحة البيانات المصغرة لقيمة عنصر التحكم. وأخيراً، يمكنك استخدام خاصية ControlSource لربط عنصر التحكم بحقل في مصدر بيانات النموذج.

لاستكشاف هذه المفاهيم سوف يتم تعديل مربع تحرير وسرد SupplierID في نموذج Products لعرض الحقول الإضافية من جدول Suppliers.

١- افتح نموذج Products في طريقة عرض Design وانقر مربع تحرير وسرد SupplierID. يوضح الشكل ٤-١٩ خصائص مربع التحرير والسرد. يشير إعداد خاصية ControlSource إلى انضمام مربع التحرير والسرد إلى حقل SupplierID في جدول Products. يعتبر إعداد خاصية RowSource هو عبارة SQL التي تقوم باسترجاع الصفوف التي تحتوي على SupplierID وcompanyName من جدول Suppliers. يشير إعداد خاصية BoundColumn إلى أنه يتم الاحتفاظ بالقيمة الموجودة في العمود الأول للصف المحدد في مربع التحرير والسرد. تشير خاصية ColumnWidths إلى أنه تم إخفاء SupplierID وعرض قيم CompanyName في مربع التحرير والسرد.

٢- انقر زر Build على يمين مربع خاصية لعرض Query Builder. اسحب حقلي ContactName وPhone إلى شبكة التصميم "انظر شكل ٤-١٩". اغلق Query Builder ثم احفظ التغييرات.



الشكل ٤-١٩ تشير خصائص مربع التحرير والسرد إلى SupplierID أنه يتم الاحتفاظ بالعمود الأول في عنصر التحكم ويتم عرض العمود الثاني في قائمة مربع التحرير والسرد "أ". أضف أعمدة ContactName وphone إلى مصدر صف مربع التحرير والسرد "ب".

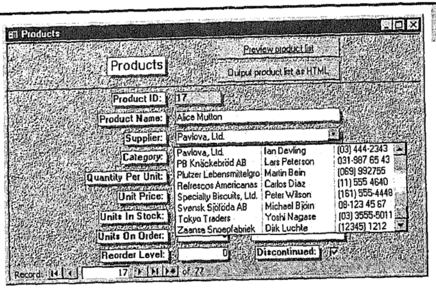
٣- قم بتغيير خصائص مربع التحرير والسرد لعرض الحقول الإضافية كصفوف صفحة البيانات المصغرة كما هو موضح بأسفل:

ColumnCount 4

ColumnWidths 0";1.2";1";0.5"

ListWidth 3.2

٤- احفظ النموذج وانتقل إلى طريقة عرض Form ثم ضع قائمة مربع تحرير وسرد Supplier "انظر الشكل ٤-٢٠". تعرض قائمة مربع التحرير والسرد ثلاثة أعمدة من جدول Suppliers كصفحة بيانات مصغرة.



الشكل ٤-٢٠
يستند نموذج Products إلى جدول Products وتعرض قائمة مربع تحرير وسرد Suppliers صفحة بيانات مصغرة تستند إلى جدول Suppliers.

يمكنك عرض القيمة من أي عمود لمربع التحرير والسرد "أو مربع القائمة" في عنصر تحكم آخر في نموذج بأي من الطريقتين: يمكنك سحب القيمة في عنصر تحكم محسوب باستخدام خاصية ControlSource أو يمكنك دفع القيمة في عنصر تحكم منضم أو غير منضم باستخدام إجراء VBA لتعيين قيمة عنصر التحكم. في أي الحالات، يتم استخدام خاصية Column لمربع التحرير والسرد أو لمربع القائمة للإشارة إلى عمود معين أو إلى عمود وصف معين.

سحب البيانات من قائمة مربع تحرير وسرد إلى عنصر تحكم غير منضم

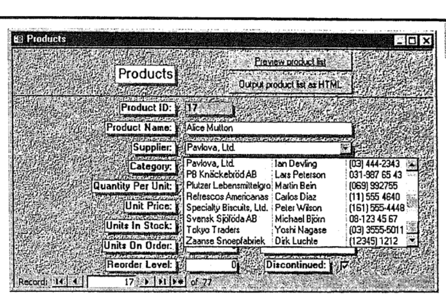
لتوضيح هذه التقنية، سيتم سحب اسم الاتصال ومعلومات الهاتف إلى عناصر التحكم غير المنضمة في نموذج Products باستخدام خاصية ControlSource:

١- افتح نموذج Products في طريقة عرض Design. وقم بوضع عنصري تحكم مربع نص غير منضمين في نموذج Products ثم أطلق على واحد منهما اسم txtContactName وزوده بخاصية ControlSource وهي =SupplierID.Column(2). ثم قم بتسمية الآخر بما يلي txtPhone وزوده بخاصية ControlSource وهي =SupplierID.Column(3). "عندما يتم إدخال تعبيرات ControlSource، سيشمل أكسس كل كلمة في أقواس مربعة". تستند خاصية Column إلى صفر مما يعني أن Column (0) يشير إلى العمود الأول بينما يشير Column (1) إلى العمود الثاني وهكذا..ثم احفظ النموذج وانتقل إلى طريقة عرض Form.

٢- استعرض السجلات. يتم تحديث عناصر التحكم غير المنضمة آلياً لعرض المعلومات باستخدام مربع التحرير والسرد "انظر الشكل ٤-٢١". تسحب عناصر التحكم غير

المنضمة الموجودة في النموذج الذي يشير إلى الأعمدة الموجودة في مربع التحرير والسرد، القيم من مربع التحرير والسرد متى يتم تغييره. وغالباً ما تعتبر عناصر التحكم المحسوبة للقراءة فقط، لا يمكنك تعديل القيم التي تم سحبها.

الشكل ٤-٢١ استخدام عناصر التحكم غير المنضمة لسحب البيانات من الأعمدة المعروضة في قائمة مربع التحرير والسرد.



يمكنك أيضاً الإشارة إلى القيم الموجودة في عمود أو أكثر في تعبير. على سبيل المثال، يمكنك وضع المعلومات في تسلسل في عمودين من صفحة البيانات المصغرة وعرض النتيجة في مربع نص فردي غير منضم بتعيين خاصية ControlSource.

٣- ارجع إلى طريقة عرض Design. احذف مربعي نص txtContactName وtxtPhone ثم ضع مربع نص فردي غير منضم في النموذج وقم بتسميته txtContactInf وزوده بخاصية ControlSource التالية:

=SupplierID.Column(2) & ": " & SupplierID.Column(3)

٤- انتقل إلى طريقة عرض Form ثم استعرض السجلات.

بشكل افتراضي، تحتوي القيم التي يتم عرضها في أعمدة مربع التحرير والسرد على نوع بيانات Text. قبل أن تتمكن من استخدام قيمة في الحسابات، قد ترغب في تحويل نوع البيانات من Text إلى نوع بيانات آخر باستخدام أحد دوال التحويل في الجدول ٤-٥.

الجدول ٤-٥: دوال تحويل البيانات

الدالة	تحويل السلسلة أو التعبير الرقمي إلى
Ccur	Currency
CDbl	Double
Cint	Integer
CLng	Long
CSng	Single
Cbool	Boolean، إذا كان التعبير صفر، يتم إرجاع False وإلا يتم إرجاع True.
Cbyte	Byte
Cdate	Date
CStr	String
Cvar	Variant
CVErr	Variant of subtype Error "متغير خطأ من نوع ثانوي".
Fix	تقوم بإرجاع الجزء الصحيح للرقم وتقريب الأعداد السالبة إلى أعلى.
Int	تقوم بإرجاع الجزء الصحيح للرقم وتقريب الأعداد السالبة إلى أعلى.

دفع البيانات من قائمة مربع تحرير وسرد في عنصر تحكم منضم

نظراً لأن التعبير الموجود في خاصية ControlSource لعنصر التحكم هو الذي يقوم بسحب القيمة من عمود مربع تحرير وسرد إلى عنصر التحكم، تعمل طريقة السحب فقط لعناصر التحكم غير المنضمة. يحتوي عنصر التحكم المنضم على خاصية ControlSource التي تم تعيينها للحقل المرتبطة به، لذلك لا تتاح خاصية ControlSource لسحب القيم. عندما ترغب في ملء عنصر تحكم منضم بقيمة من عمود موجود في صفحة بيانات مصغرة لمربع تحرير وسرد أو لمربع قائمة، فمن المفترض إنشاء إجراء VBA لدفع البيانات في عنصر التحكم. على سبيل

المثال، يستخدم مربع تحرير وسرد Bill To في نموذج Orders في Northwind، إجراء VBA لدفع CustomerName المحدد في مربع التحرير والسرد في مربع نص ShipName وبيانات العنوان في عناصر تحكم عناوين الشحن. في الفصول اللاحقة، سوف تتعرف على كيفية دفع البيانات من قائمة مربع تحرير وسرد في عنصر تحكم.

تزامن النماذج

يركز هذا المقطع على الاتصال بين عناصر التحكم الموجودة في نموذجين مفتوحين. عندما يتم فتح نموذجين بمصادر بيانات مختلفة، يتم ربط النماذج إذا كان هناك علاقة بين الجداول أو طرق العرض أو الاستعلامات الضمنية الخاصة بهذه النماذج. على سبيل المثال، يتم ربط نموذجي Customers وorders نظراً لاحتواء جداولهم الضمنية وجدولي Customers وorders على علاقة واحد بمتعدد.

عند فتح نموذجين مرتبطين، يعرض كل نموذج منهما السجل الأول في مجموعة السجلات الخاصة به ولا تتزامن النماذج. يعني تزامن نموذجين مرتبطين عرض السجلات المرتبطة في كلا النموذجين.

يمكن لأي نموذج تنظيم التزامن. على سبيل المثال، إذا كان نموذج Customers يقوم بتنظيم التزامن سيعني تزامن النموذج حينئذ تصفية نموذج Orders لعرض الطلبات المطابقة للعميل المعروض حالياً في نموذج Customers فقط. يعني الحفاظ على تزامن النماذج الحفاظ على تحديث التصفية حتى يتم تغيير نموذج Orders آلياً لعرض السجلات المطابقة فقط عندما تستعرض عميل آخر في نموذج Customers. ومن ناحية أخرى، إذا قام نموذج Orders بتنظيم التزامن، يعني تزامن النماذج بحث سجل العميل المطابق للطلب المعروض في نموذج Orders. يعني الحفاظ على النماذج متزامنة، تحديث إجراء بحث العميل ليقوم نموذج Customers دائماً بعرض العميل المرتبط عند استعراض طلب مختلف.

عندما يتم عرض نموذجين في إطارين منفصلين، تتم أفضل معالجة للحفاظ على مزامنة النماذج من خلال البرمجة. في الفصول اللاحقة، ستتعرف على تقنيات التزامن في برمجة VBA. أما الآن، سيتم استخدام Form Wizard المحترف لتزامن نموذجين.

يستخدم Form Wizard تقنيتين مختلفتين لمزامنة النماذج المرتبطة بالاستناد إلى عرض النماذج في إطارين منفصلين أو في إطار فردي. عندما ترغب في عرض النماذج في إطارين منفصلين، يستخدم Form Wizard إجراءات VBA لتزامن وإعادة تزامن نموذجين مرتبطين. عندما يتم عرض النماذج في إطار منفصل كخليط من النموذج الأساسي/والنموذج الثانوي، يعالج أكسس التزامن داخلياً وكل ما تحتاج إلى القيام به "أو قيام المعالج بتنفيذه" هو إعداد خصائص الربط.

استخدام معالج النموذج لمزامنة نموذجين

يمكنك استخدام Form Wizard لإنشاء نماذج منفصلة لمصدري بيانات مرتبط ولكتابة الإجراءات التي تحافظ على تزامن النماذج. وللتعرف على كيفية تشغيل المعالج، سوف نستخدمه لإنشاء النماذج المتزامنة لجدولي Customers أو Orders في قاعدة بيانات Northwind_ch4.

١- ابدأ تشغيل Form Wizard بنقر زر Forms في إطار Database ونقر Create Form نقراً مزدوجاً بواسطة Using Wizard.

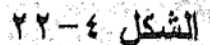
٢- في الشاشة الأولى، يتم تحديد الحقول من الجداول أو الاستعلامات المطلوبة في النموذجين "انظر الشكل ٤-٢٢ أ". حدد بعض الحقول من جدول Customers ثم من جدول Orders.

٣- تتيح لك الشاشة الثانية فرصة تعيين النموذج الذي يقوم بتنظيم التزامن وتحديد ما إذا كانت النتيجة هي نموذج يحتوي على نموذج ثانوي أو نماذج متصلة في أطر منفصلة "انظر الشكل ٤-٢٢ ب". حدد الخيار الثاني وهو Linked Forms .

٤- توفر الشاشة التالية مجموعة من الأنماط للنماذج. حدد نمط Standard.

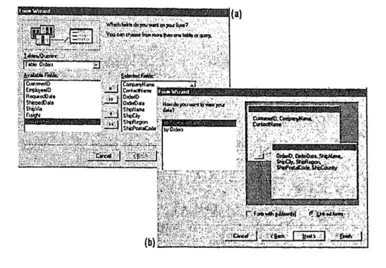
٥- يتم استخدام الشاشة الأخيرة لتحديد العناوين للنماذج المتصلة. ادخل CustomersLinked وordersLinked للعناوين ثم انقر زر Finish.

يقوم Form Wizard بإنشاء النماذج وعرض نموذج CustomersLinked "انظر الشكل ٤-٢٣ أ". يقوم نموذج CustomersLinked بتنظيم التزامن. عندما تنقر مفتاح التبديل في نموذج التحكم، يتم فتح نموذج OrdersLinked ليعرض السجلات المتطابقة. يشير اليسار الأسفل للنموذج إلى أنه يتم استخدام تصفية لتحديد السجلات "انظر الشكل ٤-٢٣". عندما تستعرض سجل آخر في نموذج CustomersLinked، يتم تغيير التصفية الموجودة في نموذج CustomersLinked، يتم تغيير التصفية الموجودة في نموذج OrdersLinked آلياً ويتم إعادة تطبيقها للحفاظ على مزامنة السجلات.



الشكل ٤-٢٢

يتم استخدام Form Wizard لإنشاء ونماذج النماذج المرتبطة.



الشكل ٤-٢٣

يقوم Form Wizard بإنشاء مفتاح تبديل في النموذج الذي يساعد على تنظيم التزامن "أ". يؤدي نقر المفتاح إلى فتح النموذج الثاني وتصفية سجلاته "ب".

CustomersLinked

OrdersLinked

Customer ID: ALFKI
Company Name: Alfreds Futterkiste
Contact Name: Maria Anders
Record: 1 of 91

OrdersLinked

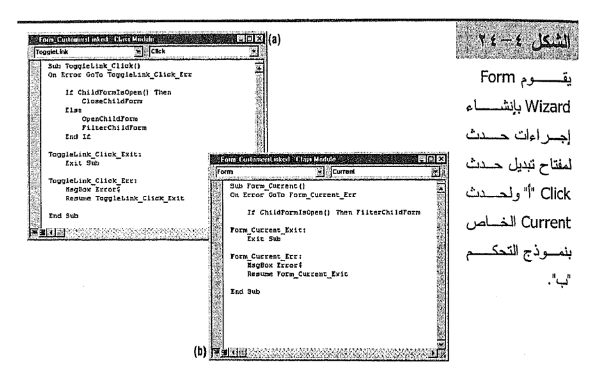
Order ID	Order Date	Ship Name	Ship City	Ship Regio	Ship Postal C	Ship Country
10643	25-Sep-95	Alfreds Futterkiste	Berlin		12209	Germany
10692	03-Nov-95	Alfred's Futterkiste	Berlin		12209	Germany
10702	13-Nov-95	Alfred's Futterkiste	Berlin		12209	Germany
10835	15-Feb-96	Alfred's Futterkiste	Berlin		12209	Germany
10952	15-Apr-96	Alfred's Futterkiste	Berlin		12209	Germany
11011	09-May-96	Alfred's Futterkiste	Berlin		12209	Germany
(AutoNumber)						

Record: 1 of 6 (Filtered)

يستمر ضغط مفتاح التبديل في نموذج CustomersLinked طوال فتح نموذج OrdersLinked. عندما تنقر مفتاح التبديل المضغوط، يتم إغلاق نموذج OrdersLinked ويعود مفتاح التبديل إلى حالة عدم الضغط. يستخدم المعالج حدثين على الأقل لتشغيل كل برنامج:

- فيستخدم حدث Click لمفتاح التبديل "انظر الشكل ٤-٢٤ أ". عندما يتم نقر مفتاح التبديل، تتحدد حالة نموذج OrdersLinked سواء كان مفتوحاً أو مغلقاً بالاستناد إلى حالة النموذج عند نقر المفتاح. إذا كان نموذج OrdersLinked مفتوحاً بالفعل، سيؤدي نقر المفتاح إلى إغلاقه، أما إذا كان مغلقاً، فسيؤدي نقر المفتاح إلى فتحه كما سيؤدي إلى إنشاء وتطبيق تصفية لتحديد السجلات المرتبطة.

- يستخدم حدث Current لنموذج CustomersLinked "انظر الشكل ٤-٢٤ ب". عند استعراض عميل آخر، يتعرف النموذج على حدث Current عندما يتم فتح النموذج لأول مرة ولكن قبل عرض السجل الأول ومتى تم إعادة استعلام النموذج". إذا تم إغلاق نموذج OrdersLinked، لا يحدث شيئاً، بينما إذا تم فتحه، يتم إنشاء تصفية جديدة وتطبيقها على نموذج OrdersLinked لتحديد السجلات الصحيحة.



الشكل ٤-٢٤

يقوم Form Wizard بإنشاء إجراءات حدث لمفتاح تبديل حدث Click "أ" ولحدث Current الخاص بنموذج التحكم "ب".

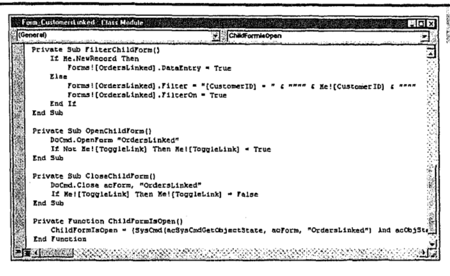
يقوم Form Wizard أيضاً بإنشاء أربع إجراءات أخرى تستخدمها إجراءات الحدث السابقة "يسمى الإجراء الذي يتم استخدامه بواسطة إجراء آخر إجراء دعم".

ستدرك بعد العمل مع النموذجين، أنه يوجد أحداث إضافية تقوم بتشغيل البرامج. نظراً لفصل النموذجين، يمكنك فتح وإغلاق كل نموذج بشكل مستقل عن الآخر. يمكنك فتح نموذج OrdersLinked بنقر مفتاح التبديل الموجود في نموذج CustomersLinked أو فتحه مباشرة من إطار قاعدة البيانات. بمجرد فتحه، يمكن أن يتم إغلاق نموذج OrdersLinked بنقر مفتاح التبديل الموجود فيه أو يمكن إغلاقه بنقر زر Close الافتراضي. أياً كان التسلسل، لاحظ تغيير شكل مفتاح التبديل آلياً لتوضيح حالة نموذج CustomersLinked. لا تهم كيفية فتح نموذج OrdersLinked حيث أن مفتاح التبديل مضغوط ولا تهم كيفية إغلاقه حيث لا يظل مفتاح التبديل مضغوط. يشير سلوك مفتاح التبديل إلى أنه من المفترض للمعالج استخدام أحداث لتشغيل البرامج التي تغير حالة مفتاح التبديل. يعتبر الحدثان الإضافيان هما حدثي Load وunload اللذان تمكن نموذج OrdersLinked من التعرف عليهما. يوضح الشكل ٤-٢٦ إجراءات الحدث

لكل من هذه الأحداث، كما يوضح الشكل ٤-٢٧ إجراء الدعم الذي تستخدمه هذه الأحداث. ستتعرف في الفصول اللاحقة على كيفية كتابة مثل هذه البرامج.

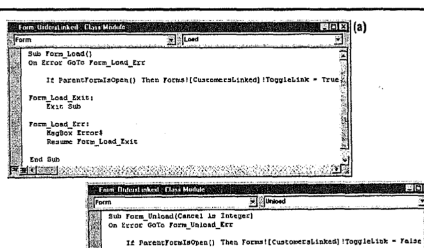
الشكل ٤-٢٥

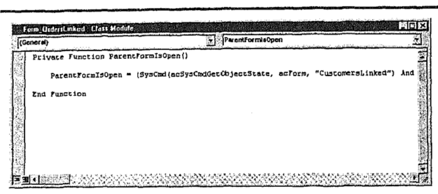
يقوم Form Wizard بإنشاء أربع إجراءات دعم يتم استخدامها بواسطة إجراءات الحدث.



الشكل ٤-٢٦

يقوم Form Wizard بإنشاء إجراءات الحدث لحدث Load "أ" وحدث Unload "ب".




```
Private Function ParentFormIsOpen()
    ParentFormIsOpen = (SysCmd(acSysCmdGetObjectState, acForm, "CustomersLinked") And
End Function
```

الشكل ٤-٢٧

يقوم Form بإنشاء Wizard إجراء دعم يتم استخدامه بواسطة كل من إجراءات الحدث التي تحافظ على تزامن مفتاح التبديل.

استخدام تقنية النموذج/النموذج الفرعي لمزامنة النماذج

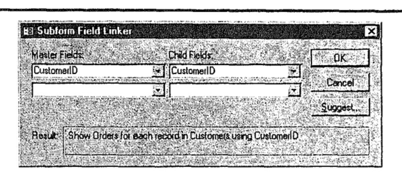
يمكنك استخدام تقنية النموذج/النموذج الفرعي لعرض النموذجين في إطار فردي. عندما تستخدم هذه التقنية لعرض المعلومات في نماذج مرتبطة، يقوم أكسس بمعالجة التزامن وإعادته.

خصائص عنصر تحكم النموذج الفرعي

يوفر أكسس عنصر تحكم نموذج فرعي/تقرير فرعي كطريقة لمزامنة نموذجين "أو تقريرين أو تقرير ونموذج" والحفاظ على سجلاتهما متزامنة. يصبح أحد النموذجين نموذج أساسي عند وضع عنصر تحكم نموذج فرعي فيه. تعتبر الثلاث خصائص الهامة لعنصر تحكم النموذج الفرعي هي SourceObject وLinkChildFields وLinkMasterFields. يتصل النموذج الثاني بالأول عند تعيين خاصية SourceObject لعنصر تحكم النموذج الفرعي إلى اسم النموذج الثاني الذي يصبح حينئذ نموذج فرعي. عند فتح النموذج الأساسي في طريقة عرض Form، يعرض عنصر تحكم النموذج الفرعي النموذج الثاني. إذا تركت خاصتي ربط السجل وهما LinkChildFields وlinkMasterFields، يتم ربط النماذج بدون سجلاتها حيث يمكنك استعراضها على نحو مستقل. لا تحتاج الجداول أو الاستعلامات الضمنية للنموذجين إلى الربط حتى يتم عرضهما في الإطار الفردي لترتيب النموذج/النموذج الفرعي.

إذا تم ربط مصادر بيانات النموذجين، يمكنك ربط السجلات بواسطة تعيين خاصتي ربط السجل لعنصر تحكم النموذج الفرعي إلى الحقول المناسبة. إذا تم ربط النموذجين في حقول متعددة، ادخل أسماء الحقول مفصولة بفواصل منقوطة في مربعات الخصائص، تأكد من إدخال الحقول المناسبة في الطلب المطابق. يمكنك استخدام Subform/Subreport Linker في تعيين خصائص ربط السجل. لبدء تشغيل Linker، افتح النموذج الأساسي في طريقة عرض Design

واعرض ورقة الخصائص لعنصر تحكم النموذج الفرعي ثم انقر زر Build على يمين أي من خصائص ربط السجل. يوضح الشكل ٤-٢٨ حوار Subform Field Linker حيث يتم تعيين حقول الربط التي لا يتم تضمينها في أي من النموذج الأساسي أو الفرعي كعناصر تحكم. بعد تعيين خصائص ربط السجلين، يتم ربط السجلات وهي متزامنة وتظل كذلك. يعرض أكسس السجلات المتزامنة في النموذج الثاني في أثناء استعراض سجلات النموذج الأساسي.



الشكل ٤-٢٨ استخدام Subform/Subreport Linker لتعيين خصائص ربط السجل.

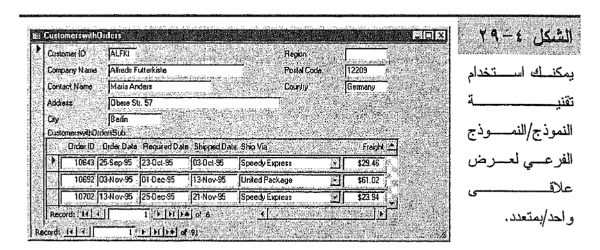
تتطلب تقنية النموذج/النموذج الفرعي تعيين خصائص عنصر تحكم النموذج الفرعي وسيحافظ أكسس آلياً على تزامن النماذج. وسيقوم أكسس بتعيين خصائص عنصر تحكم نموذج فرعي بموجب شروط معينة. إذا تم إنشاء نموذج فرعي أو تقرير فرعي بواسطة سحب نموذج أو تقرير من إطار Database إلى نموذج أو تقرير آخر، سيقوم أكسس بعرض الكائن الذي تم سحبه داخل عنصر تحكم النموذج الفرعي/التقرير الفرعي وتعيين خصائص ربط السجل آلياً إذا تم الوفاء بالشرطين التاليين:

- استناد التقرير أو النموذج الأساسي إلى جدول يحتوي على مفتاح أساسي. يقوم أكسس بتعيين خاصية LinkMasterFields إلى حقل "أو حقول" المفتاح الأساسي.
- استناد النموذج أو التقرير الفرعي إلى استعلام أو جدول يحتوي على حقل "أو حقول" بنفس الاسم وبنفس نوع البيانات أو بأخرى مناسبة كالمفتاح الأساسي للجدول الموجود أسفل النموذج الأساسي. يقوم أكسس بتعيين خاصية LinkChildFields إلى الحقول المطابقة المسماة من مصدر بيانات النموذج الفرعي أو التقرير الفرعي.

يمكنك استخدام تقنية النموذج/النموذج الفرعي لعرض علاقة واحد-بمتعدد بتأسيس كل من النموذج الأساسي على الجانب الواحد والنموذج الفرعي على الجانب المتعدد. على سبيل المثال، يحتوي جدولي Customers و orders على علاقة واحد-بمتعدد. وباستناد النموذج الأساسي إلى جدول Customers والنموذج الفرعي إلى جدول Orders، يعرض النموذج/النموذج الفرعي جميع الطلبات للعميل. يوضح الشكل ٤-٢٩ النموذج/النموذج فرعي CustomerswithOrders الذي يقوم Form Wizard بإنشائه.

استخدام معالج النموذج "Form Wizard" لمزامنة نموذجين فرعيين

يتمكن Form Wizard من إنشاء مجموعة بسيطة من النموذج/النموذج الفرعي "مثل النموذج الموضح في الشكل ٤-٢٩" ومجموعة معقدة من النموذج الأساسي مع نموذج فرعي متزامن ونموذج ثاني متزامن مع النموذج الفرعي الأول. يوضح نموذج CustomersOrders بنموذجين فرعيين متزامنين ويعرف الترتيب بنموذج واحد/بمتعدد/متعدد.



الشكل ٤-٢٩
يمكنك استخدام تقنية النموذج/النموذج الفرعي لعرض علاقة واحد/بمتعدد.

Customer Orders

Company Name Alfreds Futterkiste Country Germany

Click an order...

Order ID	Order Data	Required Date	Shipped Date
10643	25-Sep-95	23-Oct-95	03-Oct-95
10692	03-Nov-95	01-Dec-95	13-Nov-95
10702	13-Nov-95	25-Dec-95	21-Nov-95
10835	15-Feb-96	14-Mar-96	21-Feb-96

...to see order details.

Product Name	Unit Price	Quantity	Discount	Extended Price
Spegesild	$12.00	2	25%	$18.00
Chartreuse verte	$18.00	21	25%	$283.50
Rössle Sauerkraut	$45.60	15	25%	$513.00

Record: 1 of 91

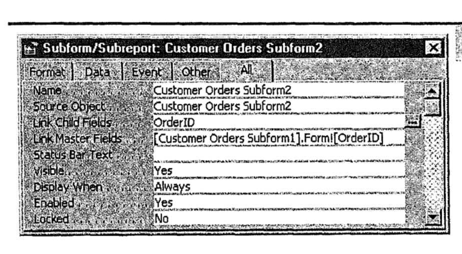
الشكل ٤-٣٠
نموذج واحد/بمتعدد/متعدد.

يعرض النموذج الأساسي سجل العميل ويعرض النموذج الفرعي الطلبات للعميل وذلك باستخدام خصائص ربط السجل لعنصر تحكم النموذج الفرعي لمزامنة السجلات في جدولي Customers وorders. وستصبح الأشياء أفضل عند مشاهدة النموذج الفرعي الثاني إلى استعلام Details Extended. يتطابق كل سجل في هذا الاستعلام مع منتج تم شراؤه في الطلب. يتم ربط النموذج الفرعي الثاني بالنموذج الفرعي الأول باستخدام OrderID كالحقل المناسب في

كل من مجموعتي السجل. يوضح الشكل ٤-٣١ إعدادات خاصية الربط المحددة في عنصر تحكم النموذج الفرعي للنموذج الفرعي الثاني، تؤدي هذه الإعدادات إلى مطابقة قيمة OrderID في النموذج الفرعي الثاني مع قيمة OrderID في النموذج الفرعي الأول.

الشكل ٤-٣١

تؤدي خصائص الربط في عنصر تحكم النموذج الفرعي الثاني إلى مزامنة النموذج الفرعي الثاني مع النموذج الفرعي الأول.



لاحظ استخدام معرف خاصية LinkMasterFields خاصية Form لعنصر تحكم النموذج الفرعي للإشارة إلى النموذج المعروض في عنصر تحكم النموذج الفرعي. "لا يتم عرض حقل OrderID في النموذج الفرعي الثاني في عنصر تحكم ولكن لا يزال في إمكانك استخدام القيم الموجودة في الحقل للربط." عندما يتم تحديد طلب في النموذج الفرعي الأول، تتم مزامنة النموذج الفرعي الثاني لعرض المنتجات التي تم شراؤها في الطلب. يوضح الشكل ٤-٣٢ هذه العلاقات.

الشكل ٤-٣٢

تشير خاصية LinkMasterFields إلى حقل OrderID.

The Form property refers to the form displayed in the subform control

[Customer Orders Subform1].Form![OrderID]

Refers to the subform control

A field in the record source of the form displayed in the subform control

إذا أردت إنشاء النموذج بدون استخدام Form Wizard، لن يتم تحديث النموذج الفرعي الثاني آلياً. عندما تحدد طلب مختلف، سيستمر النموذج الفرعي الثاني في عرض السجلات للطلب السابق. يمكنك تحديث النموذج الفرعي الثاني بتحديد طلب في النموذج الفرعي الثاني

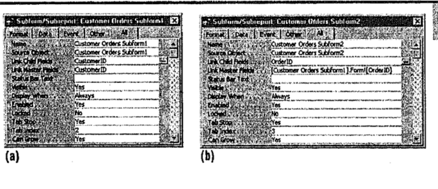
والنقر في عنصر تحكم بالنموذج الأساسي ثـــم ضغــط F9. لاحــظ أن تحديــد Records ⇦ Refresh لا يفرض تحديث النموذج الفرعي.

يتناول المعالج التحديث، عندما تستخدم Form Wizard لإنشاء النموذج. وللتعرف على كيفية قيام المعالج بتحديث التزامن، قم باتباع الخطوات التالية:

١- افتح نموذج Customer Orders في طريقة عرض Design. يساعد النموذج الأساسي على تنظيم التزامن مع النموذج الفرعي الأول. تشير ورقة الخصائص لعنصـــر تحكـم النموذج الفرعي إلى رابطة السجل المبنية علـــى قيـم CustomerID بيـن النمـوذج الأساسي ونموذج Customer Orders Subform1 "انظر الشـــكل ٤-٣٣أ". تشـير ورقة الخصائص لعنصر تحكم النموذج الفرعي الثاني "انظر الشـــكل ٤-٣٣ب" إلـى السجل الذي يربط بين النموذجين الفرعيين المبنيين على قيم OrderID.

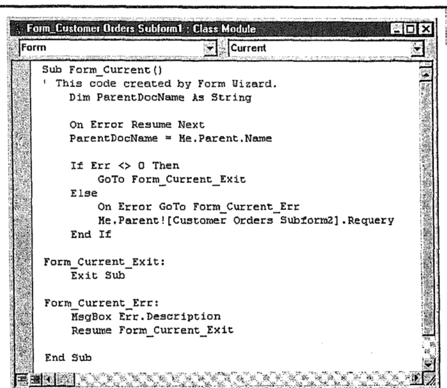
الشكل ٤-٣٣

خصـائص ربـط السجل للربط بيـن النموذج الأساسـي والنموذج الفرعـي الأول "أ" وبيـن النموذج الفرعـي الأول والثاني "ب".



١- افتح النموذج الفرعي الأول في طريقة عرض Design. تشير ورقة الخصائص لـــهذا النموذج إلى قيام Form Wizard بإنشاء إجراء الحدث الذي يتم تشغيله بواسطة حـــدث Current للنموذج.

٢- لعرض إجراء VBA، انقر زر Build على يمين مربع خاصية OnCurrent. لاحـظ أن السطر الذي يتضمن Requery يعتبر التعليمات الخاصة بإعادة استعلام النموذج الفرعي الثاني "انظر الشكل ٤-٣٤". متى يتم نقر سجل مختلف في النموذج الفرعي الأول، يقوم هذا الإجراء بتشغيل وإعادة استعلام النموذج الفرعي الثاني.




```
Sub Form_Current()
' This code created by Form Wizard.
    Dim ParentDocName As String

    On Error Resume Next
    ParentDocName = Me.Parent.Name

    If Err <> 0 Then
        GoTo Form_Current_Exit
    Else
        On Error GoTo Form_Current_Err
        Me.Parent![Customer Orders Subform2].Requery
    End If

Form_Current_Exit:
    Exit Sub

Form_Current_Err:
    MsgBox Err.Description
    Resume Form_Current_Exit

End Sub
```

الشكل ٤-٣٤ يقوم Form Wizard بإنشاء إجراء حدث من أجل حدث Current للنموذج الفرعي الثاني الذي يحافظ على النموذج الفرعي الثاني المتزامن مع النموذج الفرعي الأول.

خلاصة

تحتوي الواجهة الموجودة في معظم طلبات أكسس الآلية، على صفحات ونماذج وتقارير. وبتنفيذ آلية قاعدة بيانات، يتم كتابة إجراءات VBA التي تعتمد على فهم الاتصال بين النماذج والصفحات والبيانات الموجودة في الجداول بالإضافة إلى كيفية قيام النماذج والصفحات بتوصيل المعلومات إلى بعضها.

- يشير إعداد خاصية RecordSource للنموذج أو مقطع صفحة الوصول إلى البيانات أو التقرير إلى اتصال النموذج أو التقرير بطريقة عرض جدول أو باستعلام. يعتبر النموذج أو المقطع أو التقرير غير منضم إذا كانت خاصية RecordSource فارغة. إذا كان الإعداد هو اسم جدول أو طريقة عرض أو استعلام مخزون أو عبارة SQL، سيتم ضم النموذج أو المقطع أو التقرير إلى مصدر البيانات الضمني.
- إذا كان النموذج يحتوي على استعلام AutoLookup المستند إلى جدولين في علاقة واحد/بمتعدد كمصدر سجل، يمكنك استخدام النموذج لبحث المعلومات آلياً من جدول "واحد" عند إدخال قيمة في حقل ربط.
- يحتوي النموذج أو المقطع أو التقرير على مصدر سجل واحد ولكن توجد طرق عديدة ليحصل بها النموذج على معلومات من مصادر بيانات أخرى:

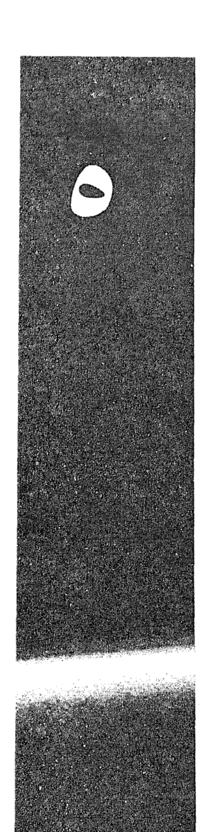
عناصر تحكم محسوبة: يتم استخدام عناصر تحكم لبحث المعلومات في نموذج أو تقرير آخر مفتوح. يمكنك استخدام دالة ()Dlookup في تعبير ControlSource لعنصر التحكم المحسوب لبحث المعلومات في أي جدول أو استعلام. تسحب تعبيرات خاصية ControlSource البيانات إلى عنصر التحكم.

البرمجة: يتم استخدام البرمجة لبحث أو حساب قيمة ثم دفعها في عنصر تحكم غير منضم "إذا احتوى عنصر التحكم على خاصية ControlSource فارغة" أو عنصر تحكم منضم.

عنصر تحكم مربع التحرير والسرد أو مربع قائمة: يمكن اتصال عنصر تحكم مربع تحرير وسرد أو مربع قائمة بجدولين أو استعلامين مختلفين: يمكن ضم عنصر التحكم إلى حقل في مصدر السجل الضمني للنموذج ويمكن أن يعرض عنصر التحكم سجلات من مصدر بيانات آخر. يمكنك استخدام عناصر التحكم هذه لتمرير القيم من جدول أو طريقة عرض أو استعلام آخر في النموذج.

- عندما لا يتم تحديث النموذج آلياً لعرض البيانات الحالية، تعتبر البرمجة ضرورية لتنفيذ التحديث آلياً.
- عندما تستند النماذج إلى الجداول أو الاستعلامات المرتبطة، يمكنك استخدام خصائص الربط المضمنة لعنصر تحكم النموذج الفرعي للحفاظ على مزامنة النماذج في ترتيب النموذج الأساسي/النموذج الفرعي.
- تعتبر البرمجة مطلوبة لمزامنة النماذج المرتبطة عندما يتم عرض النماذج في أطر منفصلة. يستخدم Form Wizard برمجة VBA للمزامنة.

يستكمل هذا الفصل الموضوعات التمهيدية التي تعد أساسية بالنسبة إلى برمجة VBA. والآن أنت على استعداد للتعرف على كيفية كتابة البرامج التي تم مناقشتها في هذه الفصول.



مبادئ برمجة VBA

يعرض هذا الفصل المبادئ الأساسية لبرمجة VBA. لكن قبل ذلك سنقوم بمراجعة بعض مفاهيم البرمجة الأساسية ثم ستتعلم بعد ذلك كيفية الحصول على وإعداد خصائص الكائن كما ستتعلم كيفية استدعاء طرق الكائن وكيفية الإشارة إلى الكائنات في برامج VBA.

ويخصص جزء ليس بقليل في هذا الفصل للحديث عن نموذج كائن تطبيق أكسس. وسنستكمل الحديث عن النموذج الذي كنا قد بدأناه في الفصل الثالث، بتركيز خاص على جوانب VBA الوحيدة وعلى الطرق الخاصة بالكائنات. وأخيراً، ستتعرف على كيفية استخدام عارض الكائن Object Browser لاستكشاف الكائنات.

مفاهيم البرمجة الأساسية

يقوم هذا الفصل بمراجعة المفاهيم الأساسية لنموذج البرمجة "أكسس" والتي ذُكرت في فصول سابقة:

- يُعرف أكسس مجموعة من الكيانات التي يمكن معالجتها ويطلق عليها اسم الكائنات "Objects".
- يُعرف أكسس مجموعة من التغييرات في الحالة، تلك التغييرات التي يتعرف عليها الكائن، ويطلق عليها أكسس اسم أحداث "events".
- يمكن معالجة الكائن وذلك بكتابة برنامج ما ثم إخبار أكسس بتشغيله عند تعرف الكائن على حدث.

وللحديث عن برمجة VBA، يجب فحص بعض المفاهيم الأخرى أولاً:

Variables: في برمجة VBA، يعد المتغير موضع تخزين مؤقت في ذاكرة تقوم أنت بتسميتها واستخدامها للاحتفاظ بقيمة ما أو للإشارة إلى كائن ما. وفي برمجة VBA يستقر المتغير في الذاكرة، ولا تظهر في واجهة المستخدم.

Objects: استخدمت كلمة كائن استخداما مطلقاً وغير رسمي لتشير إلى كيانات حقيقية في نافذة Database، لها خصائص وأحداث. والآن سنستعين بالمصطلح "كيان" في الشرح، كما سنقوم بتشكيل تعريف الكائن، باعتباره مفهوماً أساسياً ومختلفاً إلى حد ما.

Recordsets: سنرى كيف يتم ربط الشكل بالتسجيلات التي يعرضها. يوجد في برمجة VBA طريقة جديدة أساسية للعمل مع الأشكال، حيث يمكن معالجة نسخة من مجموعة تسجيلات الشكل في الذاكرة دون المساس بالتسجيلات.

ملاحظة تستخدم الأمثلة المذكورة في الفصلين ٥ و٦ تطبيق نموذج Northwind. قم بعمل نسخة من ملف Northwind.mdb يحمل اسم

الكائنات والخصائص والطرق

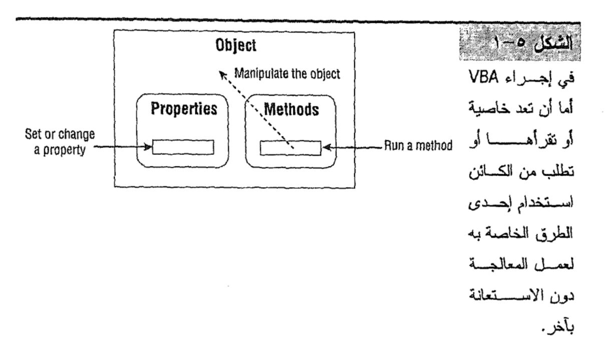
تعرض برمجة VBA عالم Access ككائنات ذات ثلاثة ميزات:

- خصائص تصف ميزات الكائن.
- طرق تصف العمليات التي يستطيع الكائن القيام بها دون الاستعانة بآخر.
- أحداث يمكن استخدامها لتشغيل البرامج "الإجراءات".

يعد كائن VBA كياناً مبرمجاً يعرف كيفية القيام بالعمليات. يمكن اعتبار الطريقة برنامجاً مصغراً أو شكلاً خطياً لإحدى العمليات التي يستطيع العصر القيام بها دون الاستعانة بآخر. وهاهي بعض النماذج التي يستطيع العنصر القيام بها دون الاستعانة بآخر. وها هي بعض النماذج لهذه الطرق: Delete وClose وEdit وQuit وMove وRequery وSave وUpdate وUndo.ويصل عدد هذه الطرق في Access VBA إلى أكثر من مائة طريقة، وهذه الطرق تكون مكتوبة من قبل ومضمنة في أشكال خطية مجهزة لاستخدامها في عمل الإجراءات اللازمة كما أنها تعد جزءاً لا يتجزأ من الكائنات.

في برمجة VBA الكائن ما هو إلا مجموعة الخصائص والطرق الخاصة به، فيقوم بقراءة الخصائص وتغييرها مباشرة "باستخدام عبارة التعيين"، ثم يطلب من الكائن استخدام إحدى الطرق للمعالجة. يوضح الشكل ٥-١ الكائن في برمجة VBA.

يتبع الأحداث تغييرات في حالة الكائن الأمر الذي يسمح بالمزيد من فرص البرمجة، حيث يمكن أن تتقاطع المعالجة التي عادة ما تتبع التغيير. ولقد تم ذكر تلك الأحداث بتفصيل أكثر في الفصل الثاني. أما الأجزاء التالية فستصف كيفية الحصول على وإعداد الخصائص، واستخدام الطرق الخاصة بالكائنات.



الشكل ٥-١

في إجراء VBA أما أن تعد خاصية أو تقرأها أو تطلب من الكائن استخدام إحدى الطرق الخاصة به لعمل المعالجة دون الاستعانة بآخر.

التغليف

تعرف عملية جمع الطرق والخصائص معاً مع الكائن باسم عملية التغليف، والتي تعد إحدى العناصر الأساسية لنموذج الكائن الاتجاهي. لا يمكن القول بأن إجراء VBA يعد لغة برمجة كائن اتجاهي لأنه مفتقر إلى إحدى العناصر الأساسية الأخرى (والتي تعرف باسم المنحة المشتركة)، بل هو كائن متمكن كما يمكن القول أيضاً بأن Access هو نظام كائن متمكن لإدارة قاعدة البيانات حيث يستخدم VBA نموذج الخصائص والطرق المخزنة معاً كوحدة واحدة في الغلاف. في هذا النموذج يفصلك الغلاف عن العالم الخارجي، حتى أنه في حالة الرغبة في معالجة الكائن يتم استدعاء إحدى الطرق الخاصة بالكائن للقيام بتلك المهمة.

على الرغم من أن الكائن هو مجموعة الخصائص والطرق، فإنه ليس كذلك في الحقيقة، فتخزين أشكال الخطوط الخاصة بكل طريقة للكائن مع تخزين الكائن نفسه ينتج عنه نوعاً من التكرار. ومن ثم سيكون المقصود من التغليف الحقيقي هو أن يكون لكل شكل في قاعدة البيانات نسخته الخاصة به من أشكال الخطوط لكل طريقة كائن Form. لكن في الحقيقة، لا تخزن الطرق الخاصة بالكائن إلا مرة واحدة مع فئة الكائن. لكن على الرغم من ذلك فإنه سيكون مفيداً اعتبار الكائن حزمة من الخصائص والطرق الخاصة به.

إعداد الخصائص

عند العمل مع خصائص في VBA يمكن اتخاذ هذين الإجراءين:

- يمكن تغيير قيمة الخاصية، أي يمكن إعداد خاصية.
- يمكن قراءة قيمة الخاصية، أي يمكن التوصل إلى إعداد خاصية

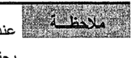
وفي كلتا الحالتين، ستشير إلى خاصية الكائن باستخدام عامل تشغيل النقطة، وذلك لفصل الإشارة إلى الكائن عن اسم الخاصية. عند إعداد خاصية يجب تعيين قيمة لها، ولإعداد خاصية استخدم علامة (=) كعلامة تشغيل للتعيين كالآتي:

```
object.propertyname = value
```

Object هي إشارة للكائن وpropertyname هي اسم الخاصية المراد إعدادها، أما Value فهو يمثل ما تتحول إليه إعداد الخاصية. فعلى سبيل المثال، هذه العبارة تقوم تغيير خاصية Caption لشكل Employees لتصبح Hi there!.

```
Forms!Employees.Caption = "Hi there!"
```

نظراً لأن قيمة الخاصية Caption تتكون من سلسلة من الأحرف، يجب وضع النص بين علامتي تنصيص. ويمكن اعتبار علامة التشغيل للتعيين ذلك السهم الذي يشير إلي اليسار، والذي يقوم بإرسال كل ما هو موجود على الجانب الأيمن لعلامة (=) إلى الجانب الأيسر، ويقوم بتخزينها في أي مكان هناك.



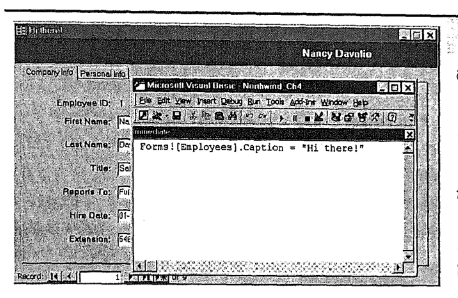
عند قراءة أو تغيير قيمة الخاصية يجب أن يكون الشكل أو التقرير الذي يحتوي على الخاصية مفتوحاً.

لقد تم التعرف على نافذة Immediate التي تستخدم لإعداد قيمة الخاصية في الفصل الثالث، فقط استخدم للكائن مرجع المعرف الكامل. وكمثال على ذلك سنقوم بتغيير خاصية Caption لشكل ما، فقط اتبع هذه الخطوات:

١- افتح نموذج Employees في عرض Form.

٢- اضغط Ctrl+G لفتح نافذة Immediate ثم اكتب Forms!Employees.Caption = "Hi there!" واضغط Enter سيعرض شريط عنوان النموذج جملة Hi there! "انظر الشكل ٥-٢". سيظل إعداد الخاصية المتغير ظاهراً طالما أن النموذج مفتوح بينما يتم تجاهله عند إغلاق النموذج.

٣- انقر النموذج وانتقل إلى عرض Design ثم انقر زر Properties ستجد أن خاصية النموذج Caption لن تتغير في ورقة الخصائص.



الشكل ٥-٢
إذا كان النموذج في عرض Form وذلك عند إعداد خاصية Caption في نافذة Immediate فسيظل إعداد الخاصية موجوداً إلى أن يتم إغلاق النموذج.

٤- انقر نافذة Immediate وضع نقطة الإدراج في أي مكان بالسطر الذي قمت بإدراجه من خلال الخطوة ٢، ثم اضغط Enter. انقر النموذج بعد ذلك سيقوم VBA حينئذ بتنفيذ السطر مرة أخرى وبتغيير خاصية Caption في ورقة الخصائص يمكن حفظ التغيير أو تجاهله.

٥- اغلق النموذج Employees دون حفظ التغيرات.

ملاحظة

إذا وجد النموذج في عرض Form عند إعداد قيمة الخاصية بطريقة مبرمجة "في إجراء VBA"، فسيكون هذا الإعداد مؤقتاً ويمكن تجاهله عند إغلاق النموذج. فإذا أردت تغيير أي خاصية مثل خاصية Caption على سبيل المثال تغييراً فعليك تغيير الإعداد في عرض Design "إما بطريقة تفاعلية أو مبرمجة" ثم حفظ التغيير وسنتناول في الفصل الثامن مدى عمر الإعدادات.

الحصول على الخصائص

عند تهيئة إعداد خاصية، فإنك تكون بذلك تقرأ الإعداد الحالي للخاصية، وستحتاج إلى الاحتفاظ بالنتيجة بمكان ما، ودائماً ما تستخدم متغيراً لهذا الغرض. وعندما يكون إعداد الخاصية هو قيمة لنص ما، فإنه يمكنك الاحتفاظ بالنتيجة وذلك بتعيين القيمة إلى متغير مستخدماً علامة (=) كعلامة تشغيل للتعيين كالآتي:

```
Let variable = object.propertyname
```

توضح كلمة Let الأساسية أن القيمة الموجودة في الجانب الأيمن لعلامة (=) قد تم تعيينها للمتغير في الجانب الأيسر، غير أن هذه الكلمة اختيارية حتى أن معظم المبرمجين لا يستخدمونها ويفضلون استخدام جملة التعيين البسيطة التالية:

```
variable = object.propertyname
```

فعلى سبيل المثال، إذا كان strSource هو اسم لمتغير متسلسل "مستخدماً علامة str لنمط التسمية المجري الذي سبق ذكره في الفصل الثاني"، فهذا يعني أن هذه الجملة هي التي تقوم بتعيين قيمة خاصة RecordSource لنموذج Employees، للمتغير strSource:

```
strSource = Forms!Employees.RecordSource
```

وبمجرد تعيين القيمة في متغير في إجراء VBA، فإنه يمكن استخدام المتغير عند العمل مع الخاصية، فعلى سبيل المثال، يمكن استخدام strSource في الإجراء بدلاً من كتابة المرجع Forms!Employees.RecordSource في كل مرة تشير إلى فيها إلى الخاصية. ولعل السبب الحقيقي لاستخدام المتغيرات في برمجة vba هو خفض عدد الأحرف عند الكتابة، وذلك باستخدام الأسماء القصيرة.

ويمكن أيضاً في برمجة VBA العمل مع الخصائص التي تمثل كائنات: فعلى سبيل المثال، تمثل خاصية، تمثل خاصية ActiveForm لكائن Screen النموذج active وعندما يشير إعداد خاصية إلى كائن فإنك تقوم بتعيين النتيجة إلى متغير الناتج مستخدماً الجملة التالية:

```
Set variable = object.propertyname
```

توضح كلمة Setأن الكائن الموجود في الجانب الأيمن يتم تعيينه في متغير الكائن على اليسار على سبيل المثال تعين الجملة التالية active form في متغير الكائن frm:

```
Set frm = Screen.ActiveForm
```

وكلمة Set هنا ليست اختيارية.

عندما يكون إعداد الخاصية قيمة نص، يمكن عرض القيمة في نافذة Immediate. وذلك بإدخال علامة استفهام (?) متبوعة بمرجع إلى الخاصية. تقوم نافذة Immediate بطباعة النص

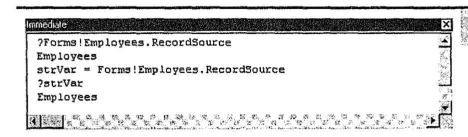
فقط، وفي حالة إشارة إعداد الخاصية إلى كائن، فإن استخدام syntax ? object.property سينتج عنه خطأ Invalid use of property" (استخدام غير صحيح للخاصية).

لمعرفة طريقة تهيئة خاصية نص، اتبع الخطوات التالية:

١- افتح نموذج Employees في عرض Form اضغط Ctrl+G لفتح نافذة Immediate، ثم اكتب ?Forms!Employees.RecordSource واضغط Enter. ستظهر كلمة Employees في السطر التالي مشيرة إلى أن جدول Employees هو RecordSource لنموذج Employees يمكن إنشاء متغير للاحتفاظ بقيمة نص، كما يمكن تعيين قيمة تعبير في نافذة Immediate. وإحدى طرق إنشاء متغير في نافذة Immediate تكون فقط بكتابة الاسم.

٢- اكتب strVar = Forms!Employees.RecordSource واضغط Enter وسيقوم VBAبإنشاء متغير strVar وبتعيين قيمة له.

٣- اكتب strVar ? واضغط Enter (انظر الشكل ٥-٣)


```
?Forms!Employees.RecordSource
Employees
strVar = Forms!Employees.RecordSource
?strVar
Employees
```

الشكل ٥-٣
يمكن استخدام نافذة IMMEDIATE لإعداد وتهيئة الخصائص.

الإعلان الضمني في مقابل الإعلان الواضح

يعرف إنشاء متغير جديد فقط باستخدامه باسم الإعلان الضمني. على الرغم من أننا نستخدم الإعلان الضمني في العمل الدراسي الذي نقوم به في الفصلين الخامس والسادس، فإننا نستخدم الإعلان الواضح عند بدء كتابة إجراءات VBA في الفصل السابع. عند السماح بإنشاء متغيرات فقط باستخدامهم لن يمكنك وقاية عملك من الأخطاء المطبعية، فعلى سبيل المثال عند إنشاء متغير بكتابة varname ثم الإشارة بعد ذلك إلى نفس المتغير ولكن بكتابة vername على سبيل الخطأ بدلاً من varname سيقوم VBA بإعداد متغير جديد يحمل اسم vername ولن يتعرف على الخطأ الإملائي الذي وقعت فيه مما يعني أن VBA لن يستطيع الربط بين الكلمتين vername و varname.

أما عند استخدام الإعلان الواضح فإنك تقوم أولاً بتحديد الأسماء التي ستستخدمها كمتغيرات. وعند إدخال النص بعد ذلك الذي تنوي أخذه كاسم لمتغير، فإن إجراء VBA يقوم بمقارنة الإدخال مع قائمة المتغيرات المعلنة ويعرض رسالة تدل على أن هناك خطأ ما وذلك في حالة عدم إيجاد الإدخال في القائمة.

استدعاء الطرق

إذا أردت أحد الكائنات أن يقوم بتشغيل إحدى الطرق الخاصة به، قم باستدعاء الطريقة المطلوبة باستخدام مرجع للكائن ولاسم الطريقة، وافصل المرجعين بعامل التشغيل النقطة وسيكون بناء الجملة لاستدعاء الطريقة كالتالي:

```
object.method
```

فعلى سبيل المثال لكائن Form طريقة Requery التي تقوم بتحديث البيانات في مصدر النموذج أن يعيد الاستعلام عن نفسه، استدع طريقة Requery الخاصة به كالتالي:

```
Forms!formname.Requery
```

تمرير الوسائط إلى الطرق

تحتوي معظم الطرق على معلومات إضافية، تعرف باسم الوسائط وتستخدم لتحديد كيفية تنفيذ الطريقة. يعرف تحديد الوسائط على ما إذا كانت الطريقة تعيد نتيجة ما.

الطرق التي لا تعيد نتائج

عندما تكون الطريقة لا تعيد نتيجة، قم بتمرير الوسائط إليها وذلك بذكرها بعد اسم الطريقة مباشرة، واستخدم الفاصلات للفصل بين الوسائط كالتالي:

```
object.methodname argument1, argument2, ..., [argumentN]
```

بالنسبة للوسائط الاختيارية فهي توضع بين أقواس مربعة، مما يعني أن الواسطة الأخيرة في التعبير السابق اختيارية، أما في باقي الوسائط التي لا توضع بين أقواس مربعة، فهي تكون مطلوبة.

فعلى سبيل المثال، تنقل طريقة GoToPage لكائن Form التركيز ليصبح على أداة التحكم الأولى في صفحة محددة في نموذج نشط ولا تعيد أي نتيجة. وتتطلب هذه الطريقة ثلاث وسائط، فيكون بناء الجملة كالتالي:

```
form.GoToPage pagenumber[,right, down]
```

form هو مرجع للنموذج بينما pagenumber هو تعبير عددي يعد رقم صفحة صحيح للنموذج النشط وكذلك right وdown فهي تعبيرات عددية لإزاحة أفقية وعمودية من زاوية النافذة اليسرى العليا فعلى سبيل المثال، إذا كان Employees (فاصل الصفحة) هو النموذج النشط فسنقل الجملة التالية التركيز ليصبح على أداة التحكم الأولى في الصفحة التالية:

```
Forms![Employees (page break)].GoToPage 2
```

الطرق التي تعيد النتائج

في حالة إعادة الطريقة لقيمة ما، ضع قائمة الوسائط بين قوسين كالتالي:

```
object.methodname (argument1, argument2, ..., [argumentN])
```

وتعد طريقة GetOption مثلاً للطرق التي تعيد نتائج، فهي تعيد القيمة الحالية لخيار ما في مربع حوار Options إذا أردت معرفة ما إذا كان شريط المعلومات قد تم عرضه أم لا، استخدم الجملة التالية:

```
Application.GetOption ("Show Status Bar")
```

إذا تم اختيار مربع الاختيار Status Bar ستعيد طريقة GetOption القيمة True وفيما عدا ذلك ستعيد False.

ملاحظة لاختبار إحدى الطرق التي تعيد القيمة في نافذة Immediate، يجب الإتيان بأمر Print قبل الجملة فعلى سبيل المثال، لاختيار طريقة GetOption اكتب ?Application.GetOption ("Show Status Bar") وسيعيد ذلك ١- إذا كان الشرط True أو إذا كان الشرط False. وعند كتابة الإجراء أو صفر إذا كان الشرط False. وعند كتابة إجراء VBA يمكن استخدام True و-1 وFalse و0 بالتبادل.

تمرير الوسائط حسب الترتيب وحسب الاسم

يمكن تمرير الوسائط إلى طريقة بشكلين مختلفين:

حسب الترتيب: عند تمرير الوسائط حسب التمرير، فستقوم بعمل قائمة بقيم الوسائط بالترتيب المحدد في بناء الجملة للطريقة. يمكن ترك وسيطة اختيارية فارغة في وسط قائمة الوسائط، لكن يجب تضمين فاصلة الوسيطة، لكن في حالة ترك أكثر من وسيطة فارغة، لا تستخدم فاصلة بعد الوسيطة الأخيرة التي تقوم بتحديدها.

حسب الاسم: عند تمرير الوسائط حسب الاسم، قم بتحديد اسم الوسيطة متبوعاً بقيمـــة الوسيطة. يمكن ذكر الوسائط المسماه بأي ترتيب، ولن تكون هناك حاجة لإدراج أحرف نائبة للوسائط المتجاهلة.

فعلى سبيل المثال لنقل التركيز إلى الصفحة الثانية، بإزاحة عمودية تقدر بستمائة تبويب يمكن تمرير الوسائط حسب الترتيب:

```
Forms![Employees (page break)].GoToPage 2, ,600
```

الفاصلة الثانية هي حرف نائب للوسيطة اليمنى الفارغة. لترك كلتا الوســـيطتين فـــارغتين، استخدم الجملة التالية:

```
Forms![Employees (page break)].GoToPage 2
```

يمكن تمرير الوسائط المسماه بأي ترتيب كالآتي:

```
Forms![Employees (page break)].GoToPage down:= 600,
pagenumber:=2
Forms![Employees (page break)].GoToPage pagenumber:=2
```

تعيين نتيجة الطريقة

معظم الطرق لا تعيد نتائج، لكن بعضها يعيد قيم نص، وبعضها يعيـــد كائنـــات. وإذا كـــانت الطريقة شيئاً أياً كان فهذا يعني أنه يمكن تعيين نتيجة المتغير إذا كانت الطريقة تعيد قيمة نـــص، فإنه يمكن استخدام عبارة تعيين Let لتعيين القيمة لمتغير وذلك كالاتي:

```
Let variable = object.method
```

وبما أن كلمة Let تعد اختياريه، فيمكنك بدلاً من ذلك استخدام:

```
variable = object.method
```

فعلى سبيل المثال، عند تحديد ما إذا كان شريط المعلومات قد تم عرضه، يمكن تعيين النتيجة مستخدماً:

```
blnStatus = Application.GetOption ("Show Status Bar")
```

وعند إعادة الطريقة لكائن، استخدم عبارة التعيين Set لتعيين النتيجة لمتغير كائن.

```
Set variable = object.method
```

استدعاء الطرق في نافذة Immediate

يمكن استدعاء طريقة في نافذة Immediate بإدخال مرجع الكائن ومرجع اســـم الطريقـــة، وفصلهما بعامل التشغيل النقطة. ادخل بعد ذلك قائمة الوسائط بعد اسم الطريقة وضع الوســـائط

بين قوسين إذا كانت الطريقة تعيد نتيجة. يمكن تمرير الوسائط إلى الطرق بحسب الترتيب أو حسب الاسم. عندما تعيد الطريقة قيمة ما، يمكن عرض القيمة في نافذة Immediate أو تعيين القيمة لمتغير. وعندما تعيد الطريقة كائناً ما، يمكن تعيين الكائن لمتغير كائن هاهي بعض الأمثلة:

١- افتح نموذج Employees "فاصل الصفحة" في عرض Form، واكتب Forms![Employees (page break)].GoToPage 2 في نافذة Immediate ثم اضغط Enter. سيعرض النموذج صفحته الثانية.

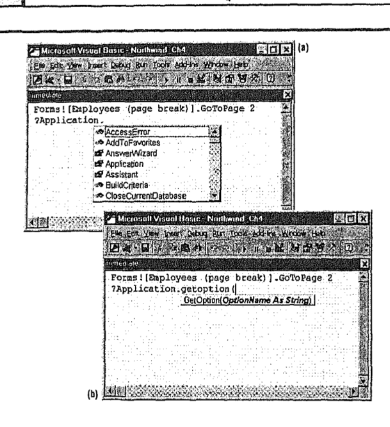
٢- اكتب ?Application.GetOption ("Show Status Bar") واضغط Enter. أثناء الكتابة ستعرض نافذة Immediate المساعدة بعد كتابة النقطة، ستعرض نافذة Immediate قائمة بخصائص وطرق كائن Application "انظر الشكل ٥-٤ أ" عند كتابة القوس الأول ستعرض نافذة Immediate بناء الجملة الخاص بطريقة GetOption لتكون دليلاً "انظر الشكل ٥-٤ ب". إذا ما تم تشغيل شريط المعلومات سيتم إعادة (True) 1–، فيما عدا ذلك سيتم إعادة (False) 0.

٣- اكتب blnStatus = Application.GetOption ("Show Status Bar") واضغط Enter سيتم إنشاء المتغير blnStatus وتعيين قيمة الخاصية له. لاختبار قيمة المتغير اكتب ?blnStatus واضغط Enter.

ملاحظة يعد استدعاء إحدى طرق الكائن إحدى وسائل معالجته فبالنسبة لكائنات Access Application يمكن تشغيل الكائنات بطريقتين إضافيتين: باستخدام طرق كائن DoCmd واستخدام الجمل والوظائف المضمنة. سيتم ذكر طرق الكائن DoCmd لاحقاً في هذا الفصل في الجزء الخاص بكائن DoCmd. ويمكن استخدام بضعة جمل ووظائف مضمنة خاصة لتنفيذ العمليات فعلى سبيل المثال، لإنشاء نموذج جديد أو تقرير أو كائنات تحكم، استخدم وظائف Create، لحذف أداة تحكم استخدم جملة DeleteControl. وسيتم الحديث عن تلك الوظائف والجمل في الفصل الرابع عشر.

الشكل ٥-٤

يمكن لنافذة Immediate أن تعرض قائمة بخصائص وطرق الكائن "أ" وبناء الجملة الخاص بالطريقة المحددة "ب".



ميزات VBA الوحيدة للتقارير والنماذج

في هذا الجزء سنلقي الضوء على بعض مميزات التقارير والنماذج التي تتوافر فقط في برمجة VBA.

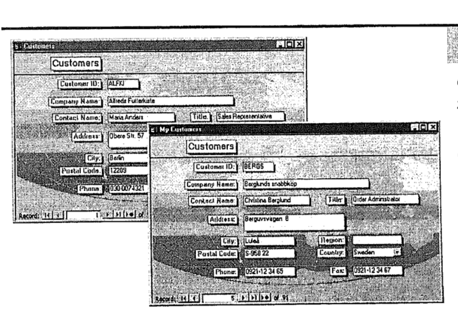
فهم مصادر السجل ومجموعات السجل

عند فتح جدول أو عرض أو عند تشغيل استعلام أو جملة SQL سيتم إعادة مجموعة من السجلات وعرضها في صفحة بيانات يمكن اعتبار مجموعة السجلات التي تم إعادتها كائناً: يمثل كائن Recordset مجموعة السجلات في جدول أو في عرض، أو التي يتم إعادتها باستعلام أو بجملة SQL.

ملاحظة تعد كائنات مجموعة السجلات إما كائنات من نوع DAO أو من نوع ADO، وقد تحتوي قاعدة البيانات على أي منهما، يديرها محرك قاعدة البيانات، أما المشروعات فيكون لها كائنات مجموعة سجلات يديرها MSDE وسيلقي الفصل الثالث الضوء على كائن مجموعة السجلات وخصائصه وطرقه.

عند تحديد خاصية RecordSource للنموذج، فإنك تكون بذلك تعين كائن Recordset محدد للنموذج. عند فتح النموذج، سيقوم محرك قاعدة البيانات بإنشاء كائن Recordset للنموذج تلقائياً، ويقوم أكسس بعرض السجلات في النموذج. قد يكون للنموذجين نفس خاصية RecordSource، لكن يكون لكل نموذج كائن Recordset الخاص به الذي ينشئه أكسس في الذاكرة.

عند فتح نموذجين لهما نفس خاصية RecordSource قد يعرضا نفس السجل، غير أنه نظراً لأن لكل نموذج مجموعة السجلات الخاصة به، فإنه يمكنك التنقل بين سجلات كل نموذج على حدة. لعمل مثال على ذلك، قم بإنشاء نسخة من نموذج Customers تحمل اسم frmMyCustomers in Northwind_Ch5,6 ويوضح الشكل ٥-٥ نموذج Customers الذي يحمل الرقم ١ كرقم السجل الحالي له كما يوضح أيضاً frmMyCustomers الذي يحمل الرقم ٥ كرقم السجل الحالي له. وطالما أن للنماذج سجلات حالية مختلفة فإنه يمكنك تحرير السجلات في كل نموذج على حدة.



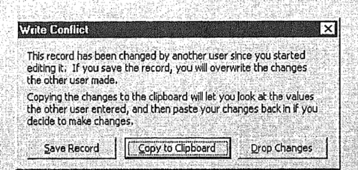
الشكل ٥-٥

نموذجان لهما نفس إعداد خاصية RecordSource لكن مجموعات سجلات مختلفة في الذاكرة.

تأمين السجل

عندما يكون للنماذج نفس السجل الحالي، فإن إجراء تأمين السجل في هذه الحالة هو الذي يقوم بتحديد التشغيل. بالافتراض لا تكون السجلات مؤمنة، ويمكن تحرير البيانات في نفس التسجيل في نموذجين في وقت واحد. في حالة حفظ التحريرات الخاصة بأحد النماذج ثم محاولة حفظ تحريرات النموذج

الآخر، ستظهر لك رسالة تعطي لك خياراً أما بكتابة فوقية للتغييرات التي أحدثتها في النموذج الأول، ونقل تسجيل النموذج الثاني إلى الحافظة، أو بتجاهل التغييرات في النموذج الثاني.



يمكن تغيير خيارات تأمين التسجيل في صفحة Advanced في مربع الحوار Options "اختر Tools ⇦ Options لعرض هذا المربع".

إنشاء مؤشر سجل حالي آخر

عند فتح نموذج يضم جدولاً أو عرضاً أو استعلاماً، سيقوم محرك قاعدة البيانات بإنشاء كائن Recordset في الذاكرة قد يعرض النموذج سجلاً واحداً أو أكثر حسب ما تتعامل معه من نموذج أوحد أو نموذج متصل أو عرض Datasheet لكن بالنسبة لمحدد السجل فهو يشير إلى سجل واحد يعرف باسم current record.

عندما تقوم بالعرض في سجل آخر مستخدماً أزرار التنقل الافتراضية الخاصة بالنموذج، يشير محدد السجل إلى السجل الذي انتقلت إليه في VBA يمكن إنشاء مؤشر سجل حالي آخر منفصل، لاستخدامه في العرض السجلات في الذاكرة بعيداً عن السجلات التي يعرضها النموذج RecordsetClone. يوضح الشكل ٥-٦ نموذجاً، ومجموعة السجلات الخاصة به، ومؤشر السجل الحالي المستقل الذي تقوم بإنشائه مستخدماً خاصية RecordsetClone.

الشكل ٥-٦

Form | Form's Recordset | Recordsetclone's current record | Form's record selector | Form's current record

النموذج، ومجموعة السجلات الخاصة به وخاصية RecordsetClone.

تعد خاصية النموذج RecordsetClone إحدى الخصائص التي تشير إلى كائن آخر، فهي تشير إلى مجموعة سجلات النموذج، لكن باستخدام مؤشر تسجيل حالي آخر، ويعرف هذا الكائن المشار إليه باسم RecordsetClone. فعلى سبيل المثال يمكن إنشاء كائن RecordsetClone لنموذج Customes. باستخدام بناء الجملة التالي:

```
Set clone = Forms!Customers.RecordsetClone
```

ولعل أحد أسباب إنشاء كائن RecordsetClone لنموذج يرجع إلى إمكانية استخدام هذا الكائن للوصول إلى معظم طرق وخصائص كائن Recordset افترض مثلاً انك تريد بطريقة مبرمجة تحديد عدد السجلات التي يعرضها النموذج. ليس لكائن Form أية خصائص يمكن أن تساعد في عملية تحديد عدد السجلات، بينما يمتلك كائن Recordset خصائص من هذا النوع، فله على سبيل المثال خاصية RecordCount التي يمكن استخدامها لتحديد عدد السجلات في مجموعة السجلات. كما يمكن أيضاً استخدامها لتحديد عدد السجلات في مجموعة السجلات كما يمكن أيضاً استخدامها لتحديد القيم الخاصة بحقل معين في السجل الحالي.

- افتح نموذج Customes في عرض Form واضغط Ctrl+G لفتح نافذة Immediate ثم اكتب ?Forms!Customers.RecordsetClone.RecordCount وأخيراً اضغط Enter. سيتم إعادة عدد السجلات في مجموعة السجلات.
- اكتب ?Forms!Customers.RecordsetClone!CustomerID واضغط Enter، سيتم إعادة القيمة في حقل CustomerID للسجل الحالي.

ملاحظة عند إنشاء نسخة من مجموعة سجلات النموذج مستخدماً خاصية RecordsetClone. لا يمكن استخدام كل خصائص وطرق كائن Recordset. على سبيل المثال، على الرغم من أن لكائن Recordset خصائص Sort وfilter التي يمكن استخدامها لفرز أو تحديد السجلات. في مجموعة السجلات، فإنه لا يمكن استخدام هذه الخصائص من النسخة حيث أنها لا تمتلك خصائص Sort أو Filter.

إن الهدف الرئيسي من استخدام خاصية RecordsetClone هو السماح بالتنقل بين أو معالجة السجلات دون المساس بالسجلات المعروضة في النموذج، ويكون ذلك باستخدام مؤشر السجل الحالي المستقل الخاص بالنسخة. ويمكن التنقل في مجموعة سجلات النموذج دون المساس

بخاصية RecordsetClone وذلك باستخدام أزرار التنقل بالنموذج وكذلك يمكن التنقل في نسخة مجموعة السجلات دون المساس بالنموذج ذاته. وباستخدام طرق Move لكائن Recordset وذلك لنقل مؤشر التسجيل الحالي للنسخة من سجل إلى آخر في مجموعة السجلات ولا نستخدم نافذة Immediate لملاحظة مؤشري السجل الحالي:

١- افتح نموذج Customers في عرض Form واختبر المراجع التالية في نافذة Immediate:

```
?Forms!Customers!CustomerID
?Forms!Customers.RecordsetClone!CustomerID
```

كلاهما يعيد حقل CustomerID الخاص بالسجل الأول.

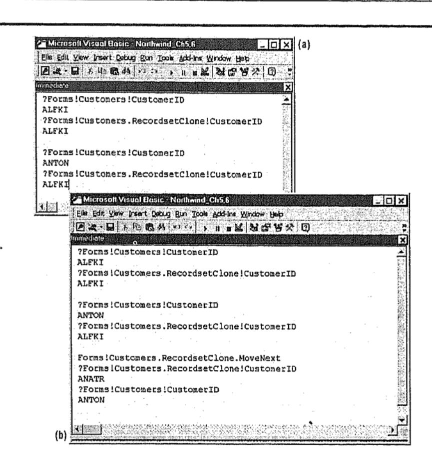
٢- قم بالعرض في سجل آخر واختبر المراجع مرة أخرى حينئذ ستشير مجموعات النموذج إلى السجل الحالي الجديد، وستظل نسخة مجموعة السجلات إلى السجل الآخر "انظر الشكل ٥-٧ أ".

٣- استدع طريقة MoveNext للنسخة في نافذة Immediate وذلك بكتابة Forms!Customers.RecordsetClone.MoveNext مع الضغط على Enter.

٤- اكتب ?Forms!Customers.RecordsetClone.CustomerID ثم اضغط على Enter ستشير النسخة حينئذ إلى السجل الثاني بينما لن يتأثر النموذج "انظر الشكل ٥-٧ ب".

ستعرف المزيد عن خصائص وطرق كائن Recordset في الفصل السادس، بما فيها طرق Move.

يمكن استخدام خاصية RecordsetClone وقتما تشاء للعمل مع مؤشري سجل حالي لمجموعة سجلات نموذج. وقد استخدمنا في الفصل الأول Combo Box Wizard لإنشاء مربع تحرير وسرد بحثي. ولقد كتب هذا المربع إجراء VBA الذي يستخدم خاصية RecordsetClone لإيجاد سجل. وسنستخدم نفس هذه التقنية في الفصل الحادي عشر لإيجاد سجل معين.



الشكل ٥-٧
يمكن التنقل في مجموعة السجلات الخاصة بالنموذج دون المساس بنسخة مجموعة السجلات "أ" يمكن استخدام طرق Move للتنقل في النسخة دون المساس بمجموعة سجلات النموذج "ب".

استخدام الإشارات المرجعية

لتعقب السجلات في مجموعة السجلات يقوم أكسس بإنشاء ما يعرف بالإشارات المرجعية والتي تعد سلسلة مزدوجة فردة يقوم أكسس بإنشائها لكل سجل في مجموعة سجلات النموذج كلما تفتح النموذج. وبالنسبة لقيمة الإشارة المرجعية للسجل، فهي تختلف عن عدد السجلات الذي يعرض في الجزء السفلي الأيسر من النموذج. وعلى أي حال، لا تعد قيمة الإشارة المرجعية أمراً مهماً حيث أنه لا يمكن عرضها أو حتى استخدامها، لكن على الرغم من ذلك، فقد تشير إلى قيمة الإشارة المرجعية للسجل مستخدماً خاصية Bookmark للنموذج، والتي تكون قيمتها قيمة الإشارة المرجعية للسجل الحالي. يمكن الاستعانة بالإشارات المرجعية للقيام بعمليتين:

- يمكن تخزين الإشارة المرجعية بل يمكن تخزين قيمتها للسجل الحالي بتعيين قيمة خاصية Bookmark لمتغير السلسلة، أي أنه يمكن حفظ مكانك في مجموعة السجلات وذلك عن طريق تخزين الإشارة المرجعية.
- يمكن إعداد الإشارة المرجعية للنموذج لقيمة مخزنة من قبل، ويمكن إعداد قيمة Bookmark للنموذج لقيمة المتغير، مما ينتج عنه عرض النموذج للسجل الأصلي.

استخدام الإشارات المرجعية للرجوع إلى السجلات

توفر الإشارات المرجعية أسرع الطرق للوصول إلى السجل. فإذا أردت الرجوع إلى السجل الحالي، قم بتخزين الإشارة المرجعية للسجل الحالي في متغير السلسلة. فعلى سبيل المثال، إذا كان strMark هو متغير السلسلة قم بتخزين الإشارة المرجعية للسجل الحالي في نموذج frmCustomers مستخدماً الجملة التالية:

```
strMark = Forms!Customers.Bookmark
```

للرجوع سريعاً إلى السجل الأصلي بعد أن تكون قد انتقلت بالفعل إلى سجل آخر، قم بإعداد خاصية Bookmark للنموذج لقيمة هذا المتغير فعلى سبيل المثال يمكن الرجوع إلى السجل الذي كنت بصدده عندما تقوم بإعداد الإشارة المرجعية مستخدماً الجملة التالية:

```
Forms!Customers.Bookmark = strMark
```

ويمكن اختبار الإشارات المرجعية في نافذة Immediate كالآتي:

١- افتح نموذج Customers واكتب جملة التعيين strMark = Forms!Customers.Bookmark في نافذة Immediate واضغط Enter ستقوم هذه بإنشاء متغير strMark ضمنياً، وبإعداد المتغير للإشارة المرجعية للسجل الحالي في نموذج Customers لاحظ رقم السجل.

٢- انتقل إلى سجل آخر. يمكن الرجوع إلى السجل الأصلي وذلك بإعداد الإشارة المرجعية للقيمة المخزنة في المتغير.

٣- اكتب جملة التعيين Forms!Customers.Bookmark = strMark واضغط Enter سيقوم النموذج بعرض السجل الأصلي.

٤- يمكن حفظ إشارات مرجعية لسجلات أخرى في مجموعة السجلات وذلك بتعيين متغيرات أخرى للإشارات المرجعية الإضافية للنموذج.

لا يتم حفظ الإشارات المرجعية مع السجلات، فعند إغلاق النموذج، ستختفي مجموعة السجلات والإشارات المرجعية، ويقوم أكسس بإنشاء مجموعة أخرى من الإشارات المرجعية في كل مرة تفتح فيها النموذج.

تخزين إشارات مرجعية لخاصية RecordsetClone

كقاعدة عامة، لا يمكن استخدام الإشارات المرجعية لمجموعات السجلات المختلفة بالتبادل، وذلك لأن لكل مجموعة سجلات مجموعة خاصة بها من الإشارات المرجعية. فعند إنشاء نسخة مجموعة سجلات، فإنك تكون بذلك تقوم بإنشاء مؤشر سجل حالي مستقل لمجموعة السجلات الخاصة بالنموذج، ومن ثم يمكن تخزين إشارة مرجعية للسجل في النسخة كالآتي:

strMark = Forms!formname.RecordsetClone.Bookmark

يمكن بعد ذلك عرض السجل في النموذج بإعداد خاصية Bookmark للنموذج للقيمة المخزنة كالآتي:

Forms!formname.Bookmark = strMark

والآن يمكن التعرف على كيفية عمل ذلك باستخدام نافذة Immediate. افتح نموذج Customers انتقل إلى مؤشر السجل الحالي في نسخة مجموعة السجلات واحفظ موضعك باستخدام إشارة مرجعية انقل بعد ذلك مؤشر السجل الحالي للنموذج إلى الموضع الذي قمت بحفظه في نسخة مجموعة السجلات وهاهي الخطوات التي يجب أن تتبعها:

١- اكتب Forms!Customers.RecordsetClone.Move 10 ثم اضغط Enter. ستقوم هذه الجملة بإنشاء النسخة وبنقل عشر سجلات إلى الأمام وبعمل السجل الذي قمت بنقله إلى السجل الحالي في النسخة، وكنوع من التوكيد اختبر قيمة ?Forms.Customers.RecordsetClone!CustomerID.

٢- اكتب strMark = Forms!Customers.RecordsetClone.Bookmark واضغط Enter.

٣- انقل النموذج إلى الموضع الذي قمت بحفظه وذلك بكتابة Forms!Customers.Bookmark = strMark ثم الضغط على Enter. للتوكيد انقر في النموذج ولاحظ أن CustomerID يطابق قيمة النسخة.

استخدام خاصية Me للإشارة إلى نموذج أو تقرير

هناك خاصية كائن وحيدة لإجراء VBA تقوم بإعداد الكائن وهي خاصية Me، التي تعد خاصية لكائنات Form وReport على حد سواء وتستخدم هذه الخاصية للإشارة إلى نموذج أو تقرير في إجراء VBA المخزن في وحدة النموذج أو التقرير النمطية.

على سبيل المثال، في الإجراءات المخزنة في وحدة النموذج النمطية لنموذج frmEmployees. يمكن إعداد خاصية Caption للنموذج باستخدام الجملة التالية:

Me.Caption = "Hi there!"

يمكن الوصول إلى خاصية RecordSource للنموذج وتخزينها في متغير strDataSource باستخدام الجملة التالية:

strDataSource= Me.RecordSource

كما يمكن تحديث مصدر بيانات النموذج باستدعاء طريقة Requery الخاصة به، كالآتي:

Me.Requery

ملاحظة لا يمكن استخدام جمل الاختبارات التي تتضمن خاصية Me في نافذة Immediate، بينما يمكن استخدام خاصية Me فقط في الإجراء المخزن في وحدة النموذج أو التقرير النمطية.

عند تشغيل إجراء أمن مخزن في وحدة النموذج أو التقرير النمطية، غالباً م يكون النموذج أو التقرير هو الكائن النشط. في هذه الحالة يمكن استخدام خاصية Me أو خاصية ActiveForm أو خاصية ActiveReport لكائن Screen للإشارة إلى النموذج أو إلى التقرير غير أن النموذج أو التقرير الذي يتم تشغيل الإجراء فيه قد لا يكون هو الكائن النشط. ونظراً لأن خاصية Me دائماً ما تشير إلى النموذج أو التقرير الذي يتم تشغيل الإجراء فيه، فإنه يجب استخدام خاصية Me بدلاً من خاصيتي ActiveForm أو ActiveReport. وذلك عند الإشارة إلى النموذج أو التقرير ذي تعليمات VBA البرمجية.

ويعد كل من كائن Screen وخاصية Me ذو أهمية عند الإشارة إلى نموذج أو تقرير دون استخدام اسمه. وعن الاختلافات بين خصائص كائن Screen وخصائص Me فهي كالآتي:

- تعتمد خاصية Me على المكان الذي أنت به عند الإشارة إلى نموذج أو تقرير، ويمكن استخدامها فقط في وحدة النموذج أو التقرير النمطية، كما أن هذه الخاصية تشير إلى النموذج أو التقرير الذي تخزن معه الوحدة النمطية. وأخيراً لا يشكل التركيز كونه على التقرير أو على النموذج أهمية بالنسبة لخاصية Me.
- يعتمد كائن Screen على ما إذا كان التركيز على الكائن المشار إليه. قد تكون بأي مكان عند استخدام كائن Screen للإشارة إلى نموذج نشط أو إلى تقرير سواء كان في إجراء أو في تعبير خاصية ControlSource لأداة تحكم غير منضمة أو في تعبير لاستعلام.

مراجع لمجموعات الكائنات

لمعالجة كائن في برنامج، يجب أولاً الإشارة إلى هذا الكائن وقد كان التركيز في الفصل الثالث على الإشارة إلى الكائن بذكر اسمه، وقد تم استخدام بناء جملة عامل التشغيل نقطة التعجب وذلك للإشارة إلى كائن ضمن مجموعة. فعلى سبيل المثال، يشير Forms!Employees!LastName إلى أداة التحكم LastName في مجموعة Controls لنموذج Employees في مجموعة Form. يمكن استخدام بناء الجملة نقطة التعجب في كل مرة ترغب في الإشارة إلى الكائن بذكر اسمه، ويكون بناء الجملة هذا مطلوب في تعبيرات الاستعلام وفي جمل SQL وفي وسائط وشروط ماكرو، وفي تعبيرات ControlSource لأدوات التحكم غير المنضمة.

يوفر أكسس VBA ثلاث طرق إضافية للإشارة إلى الكائنات الموجودة ضمن مجموعات، وتستخدم كل طريقة بناء جملة خاص موضوع بين قوسين للإشارة إلى كائن باسمه، وبالمتغير الذي يستخدم للإشارة إلى الكائن أو لفهرسته. وفي برمجة VBA، يمكن أيضاً استخدام بناء جمل موضوع بين أقواس للإشارة إلى الكائن، وبالرقم يذكر جدول ٥-١ أنواع المراجع الأربعة:

الجدول ٥-١ : أربعة طُرق للإشارة إلى كائن ضمن مجموعة

بناء الجملة	الشـرح
Collectionname!objectname	استخدم مرجع نقطة التعجب للإشارة إلى كائن ضمن مجموعة باسمه.
Collectionname("objectname	استخدم الفهرس بمرجع الاسم للإشارة إلى كائن ضمن مجموعة باسمه.
Collectionname(objectvariable)	استخدم الفهرس بمرجع المتغير للإشارة إلى كائن ضمن مجموعة وذلك باستخدام متغير السلسلة للإشارة إلى الكائن.
Collectionname(index)	استخدم الفهرس بمرجع الرقم للإشارة إلى كائن في مجموعة باستخدام رقم الفهرس المعين فيه.

ولكل بناء جملة من الأربعة السابقين ميزاته، والتي سنتعرف على بعض منها باستخدام نافذة Immediate.

استخدام الفهرس حسب الاسم

يمكن استخدام الفهرس بمرجع الاسم للإشارة إلى الكائن، وذلك باستخدام تعبير السلسلة الذي يقيم اسم الكائن. اكتب ?Forms("Customers").Caption في نافذة Immediate ثم اضغط Enter، وسيعرض إعداد خاصية Caption لنموذج Customers.

استخدام الفهرس حسب المتغير

غالباً ما تقوم بإنشاء متغير في إجراء VBA ليمثل قيمة ما، اكتب strName = "Customers" في نافذة Immediate مع الضغط على Enter لإنشاء متغير strName وقم بتعيين تعبير

السلسلة فيه. يمكن استخدام الفهرس بمرجع المتغير للإشارة إلى الكائن باستخدام المتغير اكتب ?Forms(strName) في نافذة Immediate ثم اضغط Enter، وسيعرض إعداد خاصية RecordSource للنموذج.

استخدام الفهرس حسب الموضع

بمجرد معرفة رقم موضع الكائن بالنسبة للمجموعة التي ينتمي إليها يمكن الإشارة إلى الكائن فقط باستخدام هذا الرقم. وإذا كان نموذج Customers هو أول نموذج كنت قد فتحته، فيمكن الإشارة إليه باستخدام نماذج (0) "وقم الفهرس صفر يكون مناسباً لأول نموذج تم فتحه" اكتب?Forms(0).Caption في نافذة Immediate ثم اضغط Enter وستعرض خاصية Caption للنموذج الأول في مجموعة Forms.

استخدام الفهرس حسب الرقم

ينظم أكسس كائنات المجموعة الواحدة في قائمة، وبطريقة تلقائية يعين رقم فهرس لكل كائن- ويعين كل من VBA ومحرك قاعدة البيانات أرقام الفهرس بدءاً بالرقم صفر وليس بالرقم واحد، حتى أن الفهارس الخاصة بهم أصبحت تعرف باسم الفهارس التي تسند إلى الرقم صفر (Zerobased). ويمكن إضافة كائنات وحذف كائنات أخرى من المجموعات، وستتغير مواضع كل كائن بالمجموعة عندما تتغير المجموعة ذاتها.

وسيتم تعيين مجموعة جديدة من أرقام الفهرس للكائنات في أول مرة تشير فيها إلى المجموعة. عند إضافة كائن إلى المجموعة أو إزالته، قد تتأثر مواضع الكائنات الأخرى بالقائمة وذلك لأن أكسس يقوم بتحديث أرقام الفهرس تلقائياً عندما تتغير المجموعة فمثلاً بالنسبة لمجموعة Forms، يقوم أكسس بتعيين أرقام فهرس حسب الترتيب الذي تم به تحميل النماذج، كما يقوم بتغيير أرقام الفهرس عند إلغاء تحميل النماذج، إذا كان نموذج Form هو أول النماذج التي تم فتحها فسيكون رقم الفهرس الخاص به هو 0 وستشير Forms(0) إليه. إذا قمت بفتح نموذج Customers بعد 0، فسيكون رقم الفهرس الخاص به هو ١ وستشير Forms(1) إلى Customers، وفي حالة إغلاق نموذج Employees، سيتغير رقم الفهرس الخاص به ليصبح (0) وتصبح Forms(0) هي التي تشير إليه.

ومثال آخر، يمكن القول بأن أداة التحكم الثانية في مقطع Detail لنموذج Customers هي أداة التحكم CustomerID "بينما أداة التحكم الأولى هي Label Control" يمكن استخدام أياً من المراجع التالية لأداة التحكم CustomerID:

```
Forms!Customers!CustomerID
```

```
Forms!Customers("CustomerID")
Forms!Customers(1)
```

ويمكن استخدام نافذة Immediate لاختبار هذه المراجع. فإذا كان إجراء VBA مخزناً في وحدة نموذج Customers النمطية، يمكن استخدام المراجع التالية الخاصة بأداة التحكم CustomerID في الإجراء.

```
Me!CustomerID
Me("CustomerID")
Me(1)
```

للعمل على كل كائن في المجموعة على حدة مثل تغيير خاصية مؤمنة لكل أدوات التحكم في نموذج ما فإن أفضل وسيلة تكون بالإشارة إلى أدوات التحكم بأرقام الفهرس، وبالتحليق في المجموعة، وستتعرف في الفصل التاسع على كيفية التحليق في المجموعة.

كائن تطبيق أكسس الهرمي

تعرفنا في الفصل الثالث على كائنات Application في أكسس وعلى الخصائص التي تعد مفيدة في برمجة VBA، لكن هنا ستتعرف على الكائنات والخصائص الإضافية التي تكون خير عون في مهام الأتمتة، كما ستتعرف على الطرق الخاصة بكائنات Application في أكسس.

ملاحظة يصف الفصل الثالث العديد من الخصائص التي تتوافر في برمجة VBA كما تذكر الأجزاء التالية خصائص إضافية وتذكر معظم الطرق المتوافرة، مما يعطي لك فكرة جيدة عن معدل الاحتمالات لبرامجك. غير أنه قبل استخدام أي خاصية أو طريقة في التعليمات البرمجية، يجب البحث عن تعليمات فورية تمد بكل المعلومات اللازمة عن استخدام الخاصية لبناء جملة الطريقة ولنماذج التعليمات البرمجية.

كائن التطبيق

يذكر الجدول ٥-٢ بعض خصائص كائن Application الإضافية في VBA، بينما يذكر الجدول ٥-٣ طرق كائن Application ويضم أكسس ٢٠٠٠ طرقاً للعمل مع الإنترنت مثل طريقتي AddToFavorites أو FollowHyperlink.

الجدول ٥-٢: خصائص إضافية لكائن Application

الخاصية	التشغيل	قيمة الإرجاع	الشروح
DBEngine	Read-only	DBEngine object	يشير إلى محرك قاعدة البيانات عند التحكم في أكسس باستخدام تطبيق آخر من خلال الأتمتة "Automation".
FeatureInstall	Read/write	String	يحدد كيفية التعامل مع المواقف التي يتم فيها استدعاء الطرق والخصائص التي تطلب ميزات لم تثبت بعد.
CodeContext-Object	Read-only	Object	يحدد الكائن الذي يتم تشغيل إجراء VBA فيه.
IsCompiled	Read-only	Boolean	يحدد ما إذا كانت كل الوحدات النمطية في المشروع مترجمة فإذا كانت كذلك فسيتم إعادة True.
Visible	Read/write	Boolean	يعرض أو يخفي تطبيق أكسس فإذا كانت الخاصية المرئية True فسيكون التطبيق مرئياً فيما عدا ذلك لن يكون التطبيق مرئياً عند تشغيل المستخدم لأكسس. تكون خاصية Visible هي True ولا يمكن تغييرها. لكن عند تشغيل تطبيق آخر يستخدم الأتمتة لأكسس. فإن خاصية Visible تكون False بالافتراض، ويمكن إعداد هذه الخاصية في إجراء في إجراء VBA فقط عند تشغيل تطبيق آخر يستخدم الأتمتة للتطبيق "وعندما تكون لخاصية UserControl القيمة False".
VBE	Read-only	VBE object	يمثل محرر VBA ويمكن استخدامه لإرجاع مرجع إلى كائن محرر VBA وخصائصه.

الجدول ٥-٣: طُرق كائن Application

الطريقة	الشـرح
AccessError	تقوم بإرجاع السلسلة الوصفية بسبب خطـأ أكسس أو خطـأ ADO.
AddToFavorites	تضيف الاسم أو قاعدة البيانات الحالية لمجلد Favorites
BuildCriteria	تقوم بإرجاع معيار سلسلة موزع فيظهر في خلية Criteria فـي شبكة تصميم الاستعلام، أو يظـهر كـإعداد خاصيـة Filter. استخدم هذه الطريقة لإنشاء معيار الاستعلام، أو مصفاة مستنده إلى إدخال المستخدم.
CloseCurrentDatab-ase	تقوم بإغلاق قاعدة البيانات الحالية أو المشروع "المفتوح بـالفعل في نافذة اكسس" من تطبيق آخر يتحكم في أكسس مـن خـلال الأتمتة. بعد إغلاق قاعدة البيانات المفتوحة في مثـال أكسـس الحالي يمكن إنشاء قاعدة بيانات أخرى جديده أو فتـح واحـدة أخرى موجودة بالفعل في نفس مثال أكسس
DefaultWorkspaceClone	تقوم بإنشاء كائن Workspace آخر دون الحاجة من المسـتخدم إلى تسجيل الدخول مرة أخرى. تنشئ هذه الطريقة نسخة مـن كائن Workspace الافتراضي.
Echo	تقوم بإغلاق شاشة الرسم، والتي عند إغلاقها في إجـراء VBA يجب تشغيلها مرة أخرى وإلا ستظل مغلقة حتى مـع الضغـط على Ctrl+Break وحتى لو واجه الإجراء نقطة فاصلة.
FollowHyperlink	تقوم بفتح المستند أو بفتح صفحة ويب المحـددة فـي عنـوان الارتباط التشعبي الذي تقوم بملئه باعتباره وسيطة. استخدم هـذه الطريقة لتتبع ارتباط تشعبي تملؤه أنت أو يملؤه المستخدم.
GetHiddenAttribute	تقوم بإرجاع قيمة (Boolean) سمة مستترة لكائن
GetOption	تقوم بإرجاع القيمة الحالية لخيار ما في مربع الحوار Options.
NewCurrentDatabase	تقوم بإنشاء كائن Database آخر في نافذة Access من تطبيق آخر يتحكم في اكسس من خلال الأتمتة. تضيف هذه الطريقـة

الجدول ٥-٣: طُرق كائن Application

الطريقة	الشـرح
	قاعدة البيانات لمجموعة Databases تلقائياً. فبعد إنشاء مـثال لأكسس من تطبيق آخر، يجب إنشاء قاعدة بيانات أخرى أو فتح قاعدة بيانات موجودة بالفعل.
OpenCurrentDatabase	تقوم بفتح كائن Database موجود بالفعل واعتباره قاعدة البيانات في نافذة أكسس من تطبيق آخر يتحكم في أكسس من خلال الأتمتة.
Quit	تقوم بإنهاء أكسس، فهذه الطريقة لها نفس تأثير تحديد File ⇦ Exit. استخدم وسيط الخيار تلك الخاصة بالطريقة لتحديد معالجة أي كائنات لم يتم حفظها.
RefreshDatabaseWindow	تقوم بتحديث نافذة Database بعد إضافة كائن نافذة Database أو حذفه أو إعادة تسميته.
RefreshTitleBar	تقوم بتحديث شريط العنوان بعد إعداد خصائص AppTitle أو AppIcon startup
Run	تقوم بتشغيل الإجراء الذي قمت بتحديده في قاعدة بيانات اكسس من تطبيق آخر من خلال الأتمتة أو من قاعدة بيانات اكسس أخرى.
RunCommand	تقوم بتشغيل قائمة مضمنة أو أمر شريط أدوات.
SetHiddenAttribute	تقوم بإعداد سمة مستترة لكائن.
SetOption	تقوم بإعداد القيمة الحالية لخيار ما في مربع حوار Options. استخدم الطريقتين SetOption وGetOption معاً لملاحظة وتغيير الخيارات في VBA. يمكن الحصول على وإعداد أي خيار في مربع حوار Options. غير أن التغييرات التي تحدثها تكون دائمة، لذلك لاستعادة القيم الأصلية عند إغلاق قاعدة البيانات، يجب الاحتفاظ بتلك القيم في متغيرات عامة.

طرق إعادة الاستعلام

تستخدم طرق Requery للتأكد من أن النموذج أو أداة التحكم تعرض أحدث البيانات. هناك طريقتي Requery في VBA: أولاً طريقة Requery لكائن Form أو Control، وثانياً طريقة Requery لكائن DoCmd والتي تطابق أجراء الماكرو Requery. وتقوم كلتا الطريقتين بتحديث البيانات فقط على الكائن النشط، غير أنهما يختلفان عن بعضهما البعض في نقطتين أساسيتين:

- يمكن استخدام طريقة Requery لنموذج أو أداة تحكم لتحديث البيانات وذلك عندما لا يكون النموذج هو الكائن النشط، بينما تقوم طريقة DoCmd بتحديث البيانات فقط على النموذج النشط.
- عند استخدام طريقة Requery لكائن DoCmd. يقوم أكسس بإغلاق الاستعلام أو الجدول ثم يعيد تحميله مرة أخرى من قاعدة البيانات، في حين أنه عند استخدام طريقة Requery لنموذج أو أداة تحكم، يقوم أكسس بإعادة تشغيل الاستعلام أو الجدول بدون إغلاقه أو إعادة تحميله مرة أخرى، الأمر الذي يجعل هذه الطريقة أكثر سرعة.

تقوم طريقة Requery للنموذج أو لأداة التحكم بأحد المهام التالية:

- إعادة تشغيل الاستعلام أو جملة SQL التي هي تعد مصدراً للبيانات بالنسبة للنموذج أو لأداة التحكم.
- تحديث مصدر بيانات الجدول لنموذج أو لأداة تحكم وذلك بعرض سجلات جديدة أو متغيرة وإزالة السجلات المحذوفة.
- تحديث السجلات المعروضة في النموذج، استناداً إلى التغييرات التي تحدثها لخاصية النموذج Filter.

عندما تستند أداة التحكم إلى جدول أو استعلام، وهو ما يتحقق غالباً عندما تكون أداة تحكم مربع قائمة أو مربع التحرير والسرد، أو أداة نموذج فرعي للتحكم أو أداة تحكم ActiveX أو تكون أداة تحكم محتسبة ذات تعبير ControlSource يستند إلى وظيفة إجمالية في كل هذه الحالات تقوم طريقة Requery بإعادة الاستعلام من مصدر البيانات، فيما عدا ذلك تقوم الطريقة نفسها بتنشيط بيانات أداة التحكم.

كائنات المجموعة

لكائنات المجموعة Forms وReports وModules وControls خاصيتين هما Application وCount "انظر الجدول ٣-٢ في الفصل ٣".

غير أن تلك الكائنات ليس لها طرق، لذلك يقوم أكسس بإدارتها، فعلى سبيل المثال عند فتح نموذج أو إنشاء نموذج جديد باستخدام وظيفة CreateForm في إجراء VBA، يضيف أكسس النموذج إلى مجموعة Forms تلقائياً ويقوم بتعيينها في رقم الفهرس التالي. وعند إغلاق نموذج، يقوم أكسس بإزالته تلقائياً من مجموعة Forms ويقوم بضبط أرقام الفهرس للنماذج الأخرى المفتوحة.

كائن النموذج

يشير كائن Form إلى نموذج مفتوح معين، وتعد كائنات Form ضمن كائنات مجموعة Forms "إلا في حالة فتح أو إغلاق النموذج" هناك أكثر من مائة من الخصائص الخاصة بكائن Form والتي تستخدم لتغيير شكل النموذج أو طريقته. يمكن إعداد معظم الخصائص في ورقة خصائص النموذج والتي تضم أيضاً خصائص الأحداث الاثنين والثلاثين التي يعرفها النموذج. ويذكر الجدول ٥-٤ الخصائص الإضافية لنموذج Form، بينما يوضح الجدول ٥-٥ الطرق الخاصة به.

الجدول ٥ – ٤: خصائص إضافية لكائن Form

الخاصية	التشغيل	قيمة الإرجاع	الشرح
AllowDesignChanges	Read/write	Boolean	تحدد ما إذا كان يمكن إحداث تغييرات النموذج في جميع العروض أم في عرض Design فقط.
Bookmark	Read/write	Variant array of byte data	تقوم بتخزين قيمة الإشارة المرجعية للسجل الحالي كمتغير سلسلة مزدوج منفرد قام أكسس بإنشائه لكل سجل في كل مرة يتم

الجدول ٥-٤: خصائص إضافية لكائن Form

الخاصية	التشغيل	قيمة الإرجاع	الشرح
			فيها فتح النموذج. تتوافر فقط لسجل النموذج الحالي.
DefaultContro	Read-only	Control	تقوم بإعداد الخصائص الافتراضية لنوع معين من أدوات التحكم بالنموذج.
Me	Read-only	Form object	تشير إلى النموذج نفسه عند استخدامه في خاصية Me في إجراء VBA مخزن في الوحدة النمطية للنموذج.
Module	Read-only	Boolean	يشير إلى وحدة النموذج النمطية عند الإشارة إلى خاصية Module للنموذج وخاصية HasModule معدة لتكون False سيظهر خطأ ما.
OpenArgs	Read-only	String	تحدد تعبير السلسلة المحدد كوسيطة openargs لطريقة OpenForm.
UniqueTable	Read/write	String	تقوم بتحديد جدول ليكون قد تم تحديثه عند ضم النموذج إلى عروض جداول متعددة أو إلى أجراء مخزن في مشروع أكسس.

الجدول ٥-٥: طرق كائن Form

الطريقة	الشرح
GoToPage	تنقل التركيز ليصبح على أداة التحكم الأولى على صفحة محددة للنموذج النشط.
Recalc	تقوم بتحديث كل أدوات التحكم المحتسبة بالنموذج ولهذه الطريقة نفس نتيجة الضغط عل زر F9 عندما يكون التركيز على النموذج. استخدم Recalc لتحديث أدوات التحكم المحتسبة ذات تعبيرات ControlSource المستندة إلى أدوات تحكم أخرى أو حقول في مصدر بيانات النموذج عندما يتضمن إعداد خاصية ControlSource أو وظائف SQL إجمالية، استخدم طريقة Requery.
Refresh	تقوم بتحديث السجلات في المجموعة الحالية والتغييرات التي تحدثها أنت أو غيرك للبيانات الموجودة بالفعل حيث تعدد المستخدمين. هذه الطريقة لا تغير مجموعة السجلات بحيث تجعلها تتضمن سجلات تم إضافتها، أو تجعلها تستبعد سجلات أخرى تم حذفها، حيث أن المجموعة التالية قد تم إعادة الاستعلام بها وهي لا تستبعد السجلات التي قد تغيرت والتي قد لا تعطي إجابة وافية عن الاستعلام أو تقوم بتصفية المعايير التي تم تحديدها لمصدر بيانات النموذج.
Repaint	تقوم بتحديث الشاشة وبإكمال أي إعادة حسابات خاصة بأدوات تحكم النموذج. استخدم Repaint لتحديث الشاشة وذلك في حالة تأخر إعادة الرسم أثناء تنفيذ أكسس لمهام أخرى.
Requery	تقوم بتحديث مصدر البيانات لنموذج معين يكون بناء الجملة object.Requery حيث تشير object إلى النموذج وفي حالة حذفها، تعيد الطريقة الاستعلام عن مصدر النموذج النشط.
SetFocus	تقوم بنقل التركيز ليصبح على آخر أداة التحكم كان التركيز عليها في نموذج معين. إذا لم يكن للنموذج أي أدوات تحكم متاحة، فتقوم هذه الطريقة بنقل التركيز ليصبح على النموذج

الجدول ٥-٥: طرق كائن Form

الطريقة	الشرح
	نفسه، يمكن نقل التركيز ليصبح فقط على نموذج مرئي.
Undo	تعيد تعيين نموذج كان قد تم تغييره. كل التغييرات التي تطرأ على نموذج يتم تجاهلها. وتقوم هذه الطريقة بنفس مهمة مفتاح Esc عند الضغط عليه.

كائن التقرير

يوضح الجدول ٥-٦ الخصائص الإضافية لكائن Report، بينما يوضح الجدول ٥-٧ الطرق الخاصة به.

الجدول ٥-٦: خصائص إضافية لكائن Reports

الخاصية	التشغيل	قيمة الإرجاع	الشرح
DefaultControl	Read-only	Control	تقوم بإعداد الخصائص الافتراضية لنوع معين من أدوات التحكم بالتقرير.
Me	Read-only	Report object	تشير إلى التقرير نفسه في إجراء VBA المخزن في وحدة نمطية للتقرير.
Module	Read-only	Boolean	تشير إلى وحدة تقرير نمطية، في حالة الإشارة إلى خاصية Module للتقرير، وخاصية HasModule معدة لتكون False سيظهر خطأ ما.

الجدول ٥-٦: خصائص إضافية لكائن Reports

الخاصية	التشغيل	قيمة الإرجاع	الشرح
PrintCount	Read/write	Integer	تقوم بإرجاع أو إعداد عدد مرات تقييم خاصية OnPrint للجزء الحالي من التقرير.

الجدول ٥-٧: طرق كائن Report

الطريقة	الشرح
Circle	تقوم برسم دائرة أو شكل بيضاوي أو قوس في التقرير وذلك عند ظهور حدث Print.
Line	تقوم برسم خطوط ومستطيلات في التقرير وذلك عند ظهور حدث Print.
Pset	تقوم بإعداد نقطة في التقرير بلون محدد وذلك عند ظهور حدث Print.
Scale	تقوم بتعريف نظام التنسيق لكائن Report.
TextHeight.	تقوم بإرجاع ارتفاع سلسلة النص كما ستطبع بالخط الحالي.
TextWidth	تقوم بإرجاع عرض سلسلة النص كما ستطبع بالخط الحالي.

كائن الوحدة النمطية

يشير كائن Module إلى وحدة نمطية مفتوحة معينة، وهناك نوعان لكائن Module هما الوحدة النمطية القياسية والوحدة النمطية الفئوية.

تذكر الوحدة النمطية القياسية في لوح Modules لنافذة Database وهي تستخدم لتخزين إجراءات تريد توفيرها لإجراءات أخرى، ولتخزين إجراءات وظائف تريد استخدامها كمعالجات أحداث في اكثر من نموذج أو تقرير.

أما بالنسبة للوحدة النمطية الفئوية، فهي تحتوي على تعريف للكائنات الجديدة، فيمكن إنشاء الكائن بإنشاء مثال جديد للفئة التي يعرفها النموذج. وهناك نوعان للوحدة النمطية الفئوية هما وحدات النموذج أو التقرير النمطية، والوحدات النمطية المستقلة. بالنسبة لوحدات النموذج أو التقرير النمطية، فهي تكون مخزنة ككائنات منفصلة في لوح Modules لنافذة Database. فعند فتح نموذج أو تقرير بطريقة تفاعلية أو بطريقة أو إجراء ماكرو OpenForm أو OpenReport فإنك تقوم بذلك بإنشاء النموذج أو التقرير كمثال افتراضي لوحدة النموذج أو التقرير النمطية، يمكن استخدام إجراء VBA لإنشاء أمثلة غير افتراضية لوحدات النموذج أو التقرير النمطية، أو لإنشاء كل أمثلة الوحدات النمطية الفئوية المستقلة "انظر الفصل الرابع عشر لمزيد من المعلومات عن إنشاء أمثلة غير افتراضية للوحدات الفئوية النمطية".

وبطبيعة الحال يتم إنشاء وحدات نمطية بطريقة تفاعلية من خلال الكتابة في نافذة Module، غير انه في بعض الأحيان قد تريد إنشاء أو تغيير وحدة نمطية بطريقة مبرمجة. والغرض من تعريف كائن Module هو السماح بإنشاء وتعديل الوحدات النمطية مباشرة من إجراءات VBA أخرى. ويمكن استخدام خصائص وطرق كائن Module لإنشاء إجراءات جديدة في الوحدات النمطية القياسية والفئوية وذلك لإنشاء إجراءات أحداث جديدة في وحدات النموذج أو التقرير النمطية، أو لإدراج أو استبدال أو حذف سطور من التعليمات البرمجية في الحدات النمطية.

أما الوحدات النمطية الفئوية المستقلة، فهي ميزة جديدة في أكسس ٢٠٠٠. وهذا النوع من الوحدات النمطية يسنح بتعريف كائنات جديدة لا ترتبط بنموذج أو بتقرير. ويستخدم أكسس نفسه فئات مستقلة لتعريف الكائنات فعلى سبيل المثال، تقوم فئة مضمنة بتعريف كل أدوات التحكم في مربع الأدوات، فعند إدراج أداة تحكم في نموذج أو تقرير فإنك تكون بذلك تقوم بإنشاء مثال جديد لفئة أداة التحكم.

ويمكن أيضاً استخدام الوحدات الفئوية النمطية المستقلة لتعريف الكائنات الخاصة بك. فعلى سبيل المثال، يمكن إنشاء فئة Orders باستخدام الخصائص المخزنة في جدول Orders، وباستخدام طرق حساب ضريبة المبيعات وذلك للتدقيق في مستويات الجرد ولطباعة فاتورة. ويعد كل أمر من هذه الأوامر مثالاً للفئة. بإنشاء وحدة Orders النمطية الفئوية، يمكن حزم "تغليف" كل المعلومات اللازمة عن الفئة في وحدة نمطية واحدة، مما يسهل عملية تعديل الفئة فيما بعد. ولعل سهولة التعديل هو أحد أسباب استخدام الوحدات النمطية الفئوية. لكن السبب الرئيسي هو إمكانية إنشاء وحدات نمطية فئوية أخرى يمكن إعادة استخدامها في قواعد بيانات أكسس أخرى أو غيرها. وبتوافر مجموعة من الكائنات التي يمكن إعادة استخدامها، يمكن تقليل الوقت المستخدم في عملية التطوير. كما أن إضافة وحدات نمطية فئوية مستقلة يعطي VBA في أكسس الميزات الجديدة الضرورية لإنشاء الكائنات التي يمكن إعادة استخدامها في تطوير قاعدة البيانات.

يقدم أكسس ٩٧ حدثين جديدين يمكن للوحدات النمطية الفئوية التعرف عليهما. كما تتعرف الوحدة النمطية الفئوية على حدث Initialize وذلك عند إنشاء مثال جديد للوحدة النمطية الفئوية في إجراء VBA، وتتعرف كذلك على حدث Terminate عند إزالة المثال من الذاكرة.

يوضح "الجدول ٥-٨" خصائص كائن Module، بينما يوضح "الجدول ٥-٩" الطرق الخاصة به.

الجدول ٥-٨: خصائص لكائن Module

الخاصية	التشغيل	قيمة الإرجاع	الشرح
Application	Read-Only	Set by Access	تقوم بتشغيل كائن Application النشط.
CountOfDeclarationLines	Read-Only	Long Integer	تقوم بإرجاع أرقام اسطر التعليمات البرمجية في مقطع Declarations للوحدة النمطية وتساوي القيمة رقم أول سطر يأتي بعد مقطع Declarations.
CountOfLines	Read-Only	Long Integer	تقوم بإرجاع اسطر التعليمات البرمجية في الوحدة النمطية.
Lines	Read-Only	String	تقوم بإرجاع سلسلة المحتويات لسطر واحد معين أو اكثر. فقط قم بتحديد رقم السطر الأول ورقم السطور التي تود إرجاعها.
ProcBodyLine	Read-Only	Long	تقوم بإرجاع الرقم الذي يحدد السطر يبدأ فيه إجراء

الجدول ٥-٨: خصائص لكائن Module

الخاصية	التشغيل	قيمة الإرجاع	الشرح
			الوحدة النمطية. فقط قم بتحديد نوع الإجراء وتعبير السلسلة الذي يقيم باسم الإجراء.
ProcCountLines	Read-Only	Long	تقوم بإرجاع أرقام السطور في إجراء معين فقط حدد نوع الإجراء وتعبير السلسلة الذي يقيم باسم الإجراء.
ProcOfLine	Read-Only	String	تقوم بإرجاع اسم الإجراء الذي يحتوي على سطر محدد. فقط حدد رقم السطر ونوع الإجراء.
ProcStartLine	Read-Only	Long	تقوم بإرجاع رقم السطر الذي يبدأ فيه إجراء محدد. فقط حدد نوع الإجراء واسمه "كتعبير سلسلة".
Type	Read-Only	Has the value 0 for a standard module and 1 for a class module	تحدد ما إذا كانت الوحدة النمطية قياسية أم فئوية.

الجدول ٥-٩: طرق كائن Module

الطريقة	الشروح
AddFromFile	تضيف محتويات ملف نص إلى وحدة نمطية مفتوحة. تقوم هذه الطريقة بإدراج المحتويات بعد مقطع Declarations مباشرة وقبل الإجراء الأول.
AddFromString	تضيف سلسلة إلى وحدة نمطية مفتوحة. تقوم هذه الطريقة بإدراج محتويات السلسلة بعد مقطع Declarations مباشرة وقبل الإجراء الأول.
AddFromString	تضيف سلسلة إلى وحدة نمطية مفتوحة. تقوم هذه الطريقة بإدراج محتويات السلسلة بعد مقطع Declarations مباشرة وقبل الإجراء الأول.
CreateEventProc	تقوم بإنشاء قالب التعليمات البرمجية لإجراء حدث، وبإرجاع عدد صحيح طويل موضحة رقم السطر الأول. وبعد إنشاء القالب، يمكن إضافة سطور تعليمات برمجية باستخدام طريقة ما مثل طريقة InsertLines.
DeleteLines	تحذف السطور في والحدة النمطية. فقط حدد رقم السطر الأول وأرقام السطور التي تريد حذفها.
Find	يوجد نصاً محدداً في الوحدة النمطية. فقط حدد تعبير السلسلة الذي تريد إيجاده، ومعاملي البحث بما فيها توضيح لما إذا كنت تبحث عن كلمة كاملة، أو أحف مطابقة أو تضمين بطاقات رابحة. وحدد بداية ونهاية البحث إذا تم إيجاد السلسلة، فسترجع True فيما عدا ذلك سترجع False.
InsertLines	تقوم بإدراج سطر أو مجموعة سطور في الوحدة النمطية، فقط حدد السطر لتبدأ عملية الإدراج.
InsertText	تقوم بإدراج سلسلة محددة في الوحدة النمطية. سيوضع النص المدرج في نهاية الوحدة النمطية.
ReplaceLine	تقوم باستبدال أحد السطور بسطر آخر.

كائنات التحكم "Control"

قدم الفصل الثالث العديد من خصائص التحكم المتوفرة في VBA وفي هذا الفصل نقدم طرق وخصائص VBA أخرى لكائنات Control، مثل كائن Hyperlink وكائن Page ومجموعة Pages مع أداة التحكم Tab. ويوضح "الجدولان ٥-١٠ و٥-١١" خصائص وطرق الفئات العامة لأدوات التحكم.

الجدول ٥-١٠: خصائص VBA الوحيدة لأدوات التحكم

الخاصية	التشغيل	قيمة الإرجاع	الشرح
Hyperlink	Read-Only	Hyperlink Object	تقوم بتشغيل خصائص وطرق كائن Hyperlink المقترنة بخيار زر أمر، أو تسمية أو صورة أو قائمة.
ControlType	Read/Write	Integer	تحدد أو تغير نوع أداة التحكم بالنموذج وتوجد هذه الخاصية في عرض Design للنموذج.

الجدول ٥-١١: طُرق كائنات Control

التشغيل	قيمة الإرجاع	الشرح
Setfocus	أداة تحكم يمكنها استقبال التركيز	تنقل التركيز ليصبح على أداة التحكم المحددة للنموذج النشط. يمكن نقل التركيز ليصبح فقط على أداة تحكم مرئية ومتاحة. هناك بعض خصائص أدوات التحكم يمكن إعدادها فقط عندما لا يكون لأداة التحكم تركيز مثل خصائص Visible وlocked وenabled.

الجدول ٥-١١: طُرق كائنات Control

التشغيل	قيمة الإرجاع	الشرح
Requery	أدوات تحكم للبيانات	تقوم بتحديث أداة التحكم بالنموذج. يكون بناء الجملة كالآتي Object Requery حيث تشير Object إلى أداة التحكم، وعند حذفها تعيد الطريقة الاستعلام على أداة التحم النشطة للنموذج النشط.
SizeTofit	أدوات تحكم لها حجم	تغير حجم أداة التحكم لتناسب النص أو الصورة التي يحتوي عليها النص.
Undo	أدوات تحكم لها قيمة	تعيد تشغيل أداة التحكم عند تغيير قيمتها.تقوم بنفس وظيفة مفتاح Esc عند الضغط عليه.

يمكن استخدام نافذة Immediate لاختبار هذه الطرق فعلى سبيل المثال، اتبع الخطوات التالية بعد فتح نموذج Customer في عرض Form:

١- اكتب Forms!Customers!Country.SetFocus واضغط Enter سيكون التركيز على أداة التحكم Control، وللتأكيد انقر في أي مكان بنموذج Customer ولاحظ أن Country هي أداة التحكم النشطة.

٢- اكتب Forms!Customers!Country.Requery واضغط Enter سيتم تحديث أداة التحكم Country بإعادة الاستعلام عن الجملة SQL التي هي المصدر الأساسي لمربع التحرير.

٣- اكتب Forms!Customers.Requery واضغط Enter سيتم إعادة الاستعلام عن مصدر السجل لنموذج Customer.

أدوات تحكم مربع التحرير والسرد ومربع القائمة

لمربعات التحرير والسرد ومربعات القائمة خصائص وطرق عدة تفيد في برمجة VBA. يوضح "الجدول ٥-١٢" نماذج لتلك الخصائص، بينما يوضح "الجدول ٥-١٣" نماذج للطرق.

الجدول ٥-١٢: خصائص أدوات تحكم مربعات التحرير والسرد ومربعات القائمة

الخاصية	التشغيل	قيمة الإرجاع	الشرح
ListCount	Read-Only	Integer	تحدد عدد الصفوف في مربع القائمة، أو في جزء مربع قائمة يعد جزءاً من مربع التحرير والسرد.
ListIndex	Read-Only	Integer	يحدد العنصر الذي يتم تحديده في مربع القائمة، أو في جزء مربع قائمة يعيد جزءاً من مربع التحرير والسرد.
Selected	Read/Write	Zero-based array	يحدد العنصر، أو يحدد ما إذا كان العنصر قد تم تحديده بالفعل في مربع القائمة "فقط بالنسبة لمربعات القائمة".

الجدول ٥-١٣: طرق أدوات تحكم مربع التحرير والسرد ومربع القائمة

الطريقة	الشرح
DropDown	ترغم مربع التحرير والسر على الانسدال
ItemData	يعيد البيانات في العمود المنضم لصف محدد في مربع القائمة أو مربع التحرير والسرد. ويكون بناء الجملة كالآتي Control1.ItemData بحيث تشير Control إلى مربع التحرير والسرد أو مربع القائمة. وتشير rowindex إلى الصف الذي يحتوي على البيانات التي تريد إرجاعها.تتم فهرسة الصفوف بدءاً من الرقم صفر.

يمكن استخدام نافذة Immediate لاختبر هذه الخصائص والطرق فعلى سبيل المثال، اتبع الخطوات التالية بعد فتح نموذج Customer في عرض Form:

١- اكتب Set cbo = Forms!Customers!Country واضغط Enter. تقوم هذه الجملة بإنشاء cbo كمتغير كائن ليمثل مربع التحرير والسرد. وسيستخدم هذا المربع المتغير كاختصار للمرجع بأكمله.

٢- اكتب ?cbo.ListCount واضغط Enter ستحتوي القائمة على ٢١ مدينة.

٣- اكتب ?cbo.ListIndex واضغط Enter سيتم عرض رقم الفهرس للمدينة في الصف الحالي.

٤- اكتب ?cbo.ItemData(3) واضغط Enter ستعرض نافذة Immediate البيانات في الصف الرابع. على الرغم من أنك تستخدم طريقة ItemData لإرجاع البيانات في عمود محدد "العمد المنضم" فإنه يمكنك استخدام خاصية Column لعرض البيانات في أي عمود. ولخاصية Column وسيطتان تستخدمان لتحديد الفهرس للعمود وللصف الذي يبدأ في كلتا الحالتين بالرقم صفر، تكون الوسيطة الأولى هي العمود الذي يحتوي على البيانات، بينما تكون الوسيطة الثانية الصف. عند إزالة الوسيطة الثانية، يتم إعادة البيانات للصف الحالي.

٥- اكتب ?cbo.Column(0) واضغط Enter ستعرض نافذة Immediate البيانات في العمود الأول للصف الحالي. لعرض البينات في أي صف، استخدم المعالج الثاني لخاصية Column. فعلى سبيل المثال، اكتب ?cbo.Column(0,2) واضغط Enter لعرض البيانات في الصف الثالث.

مجموعة العناصر المحددة "ItemSelected"

يمكنك مربع القائمة القياسي من تحديد عنصراً واحداً منى القائمة، ويمكنك استخدام خاصية MulteSelect لاختيار اكثر من عنصر. يحيط بالصف الحالي في مربع القائمة مربع منقط. ولخاصية MulteSelect القيم التالية:

None: لإجراء تحديد واحد فقط، حدد العنصر بنقر الصف أو باستخدام مفاتيح السهم لنقل مؤشر السجل الحالي إلى أعلى أو إلى اسفل الصف ثم حدد سجلاً حالياً جديداً. عند تحديد سجل جديد يلغى تحديد السجل الساق تلقائياً. فبعد تحديد الصف، تكون الطريقة الوحيدة لإلغاء هذا التحديد هي تحديد صف آخر.

Simple: لتحديد عناصر عديدة لكن باختيار كل عنصر على حدة حدد العنصر بنقر الصف، فبعد تحديد الصف يمكنك إلغاء تحديده وذلك بنقره مرة ثانية.

Extended: لمد تحديد عناصر متعددة، فبعد تحديد العنصر، يمكنك مد التحديد بثلاث طرق:

- الضغط باستمرار على زر الماوس الأيسر والسحب إلى صف آخر.
- الضغط على مفتاح Shift ثم نقر صف آخر.
- الضغط على مفتاح Shift ثم الضغط على مفتاح السهم.

في كل حالة من هذه الحالات الثلاث، يمكنك مد التحديد من العنصر السابق المحدد إلى العنصر الحالي.وبعد الانتهاء من مد التحديد، يمكنك تحديد أو إلغاء تحديد أي عنصر بالضغط على مفتاح Ctrl ثم نقر العنصر.

عند تحديد عناصر متعددة في القائمة، يقوم أكسس بإنشاء مجموعة تعرف باسم مجموعة ItemsSelected.كل عنصر في هذه المجموعة يمثله رقماً صحيحاً يشير إلى صف محدد في مربع القائمة أو مربع التحرير والسرد، وعلى الرغم من ذلك، فإن نوع البيانات الخاص بتلك العناصر يكون Variant بالافتراض. وليس لمجموعة ItemsSelected أي طرق، بينما لها خاصية واحدة تعرف باسم Count، وهي التي تعيد أرقام العناصر المحددة. انظر الفصل الثالث عشر لمزيد من المعلومات عن مربع القائمة متعدد التحديد.

كائن الارتباط التشعبي HyperLink

يعد التشغيل المباشر للإنترنت ميزة جديدة في أكسس ٩٧. وكما هو متوقع قام مايكروسوفت بتحسين هذه الميزة في أكسس ٢٠٠٠ بل أضاف عليها إمكانات اكثر. يقدم VBA في أكسس كائن HyperLink لمعالجة الارتباطات التشعبية في إجراء VBA ولكل من زر الأمر وأداة تحكم الصورة، وأداة تحكم التسمية خاصية HyperLink تشير إلى كائن HyperLink وتمكنك من تشغيل خصائص وطرق كائن HyperLink، بالإضافة إلى إمكانية تحويل أمر قائمة إلى ارتباط تشعبي.على الرغم من أنه يمكن إعداد خصائص HyperLinkAddress وHyperLinkSubAddress في ورقة خصائص أداة التحكم، فإنه يمكن أيضاً تحديد العنوان في VBA باستخدام خصائص Address وsubAddres لكائن HyperLink، الذي له أيضاً الطرق الموضحة في الجدول ٥-١٤ ويكون بناء الجملة لهذه الطرق كالآتي:

Object: هي مرجع لكائن أداة التحكم الذي يحتوي على الارتباط التشعبي ابحث عن اسم الطريقة في التعليمات الفورية عن بناء الجملة لوسيطة الطريقة ونماذج التعليمات المبرمجية.

الجدول ٥-١٤: طرق كائن Hyperlink

الطريقة	الشـرح
AddToFavorites	تضيف عنوان الارتباط التشعبي الموجود في أداة التحكم إلى مجلد Favorites.
Follow	تفتح المستند أو صفحة ويب التي يحددها عنوان ارتباط تشعبي مفتوح بأداة تحكم. في حالة عدم اقتران عنوان الارتباط التشعبي بأداة تحكم، استخدم طريقة التطبيق Follow HyperLink.

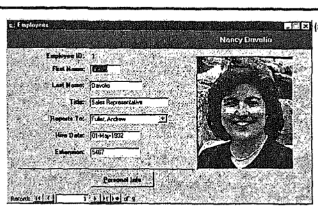
أداة تحكم علامة الجدولة"Tab"

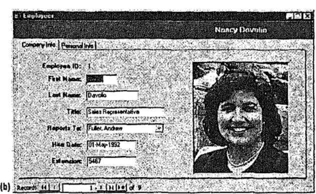
تعد أداة تحكم علامة الجدولة نموذجاً جديداً لأدوات التحكم في أكسس ٩٧، وقد يكون لها صفحة واحدة أو أكثر، كل مع علامة الجدولة الخاصة بها، ويمكن وضع أدوات تحكم نماذج أخرى في كل صفحة. فعند نقر علامة الجدولة، تصبح الصفحة التالية نشطة.

في الإصدارات السابقة لأكسس. كان يمكن الوصول إلى نفس النتيجة بإنشاء نموذجاً ذي صفحات متعددة ذات أزرار أوامر لعرض صفحات أخرى "أو كان يمكن استخدام علامة الجدولة أو أداة تحكم شريط علامة الجدولة المخصصة التي يقدمها Access DeveloperToolkit" لكن بدء من أكسس ٩٧، قامت أدوات تحكم علامات الجدولة المضمنة بجاب جعل تعدد الصفحات للنموذج اكثر وضوحاً واكثر سهولة في الاستخدام. وتحتوي Northwind على كلا النوعين للنموذج. فبالنسبة لنموذج Employee "فاصل صفحة" فهو يستخدم أداة تحكم فاصل الصفحة لتعريف صفحات منفصلة "انظر الشكل ٥-٨أ" ويستخدم نموذج Employee أداة تحكم علامة الجدولة الجديدة "انظر شكل ٥-٨ ب"

الشكل ٥-٨

يمكن عرض صفحات متعددة في نموذج واحد باستخدام أداة تحكم فاصل الصفحة "أ" أو استخدام أداة تحكم علامة الجدولة "ب"





كائن الصفحة "Page" ومجموعة الصفحات "Pages"

بطبيعة الحال، يمكن إنشاء علامة الجدولة وصفحاتها في عرض Design للنموذج، ويمكن أيضاً إنشاء، نقل أو حذف صفحات بطريقة مبرمجة باستخدام VBA. يمثل الصفحة كائن Page. وأداة تحكم علامة الجدولة مجموعة Pages تضم كل كائنات Page ومجموعة Pages تلك تبدأ أرقام فهرس من الرقم صفر ولها خاصية Count التي تستخدم لتحديد أرقام الصفحات في أداة تحكم علامة الجدولة.

ويوضح الجدول "٥-١٥" طرق مجموعة Pages

الجدول ٥-١٥: طرق مجموعة Pages

الطريقة	الشرح
Add	تضيف صفحة جديدة إلى أدلة تحكم علامة الجدولة
Remove	تحذف صفحة من أداة تحكم علامة الجدولة



لا يجب خلط عناصر مجموعة Pages مع صفحات بيانات التشغيل والتي تعد عناصر لمجموعة DataAccessPages وتستخدم Pages في هذا السياق كنوع من فهرس علامة الجدولة، بينما صفحات تشغيل البيانات هي كائن تشغيل بينات في ذاته.

لكل صفحة اسماً خاصاً بها، ويمثلها كائن Page. يمكن الإشارة إلى صفحة محددة باستخدام أي نوع من أنواع بناء الجمل الموضحة في "الجدول ٥-١" فعلى سبيل المثال، في نموذج Employee يكون اسم أداة تحكم علامة الجدولة Tapct10 ويمكن الإشارة إلى صفحة CompantInfo باستخدام المرجع التالي المهيأ لذلك:

Forms!Customers!Tabctl0.Pages!CompanyInfo

ويمكن الإشارة إلى أداة تحكم بالصفحة بالإشارة إلى مجموعة Controls للصفة ثم إلى أداة التحكم،فمثلا، للإشارة إلى أداة التحكم LastName لصفحة CompantInfo استخدام المرجع التالي:

Forms!Customers!Tabctl0.Pages!CompanyInfo.Controls!LastName

لكائن Page خاصية PageIndex التي تحدد موضعه في مجموعة Pages لأداة تحكم علامة الجدولة. ولخاصية PageIndex التشغيل read/Write كما أنها تعيد قيمة رقم صحيح. ويمكن إعداد هذه الخاصية في ورقة الخصائص أو في VBA. ويوضح "الجدول ٥-١٦طرق كائن Page.

الجدول ٥-١٦: طرق كائن Page

الطريقة	الشرح
Requery	تقوم بتحديث أدوات التحكم بالصفحة
SetFocus	تنقل التركيز بحيث يصبح على آخر أداة تحكم كان التركيز عليها في الصفحة المحددة أدوات تحكم متاحة تقوم الطريقة بنقل التركيز ليصبح على الصفحة نفسها.

كائن الشاشة "Screen"

كما أوضحنا سابقاً، يشير كائن Screen إلى نموذج أو تقرير أو أداة تحكم معينة يكون عليها التركيز في الوقت الحالي، أو يشير أداة التحكم التي كان عليها التركيز قبل ذلك. وباستخدام كائن Screen في أجراء VBA، يمكن الإشارة إلى الكائن النشط دون معرفة اسمه غير أن الإشارة إلى كائن Screen لا تجعل من النموذج أو التقرير أو أداة التحكم الكائن النشط

ولقد ذكرنا في الفصل خصائص كائن Screen، لكن هناك خاصية VBAإضافية لكائن Screen، هي MousePointer التي تحدد نوع مؤشر الماوس الذي يعرض حالياً. ولهذه الخاصية التشغيل Read/Write كما أنها تعيد رقماً صحيحاً كالآتي:

0	المؤشر الافتراضي
1	سهم
3	I-beam
7	Size N,S
9	Size E,W
11	Hourglass

وجميع خصائص كائن Screen تعيد كائنات. وعند استخدام كائن Screen في إجراء VBA، فأنك تقوم بطبيعة الحال بإنشاء متغير للإشارة إلى الكائن الذي تمت إعادته:

```
Set frmvar = Screen.ActiveForm
```

لكن ليس لكائن Screen أية طرق.

في VBA يقوم كائن Screen بدوره بالاستعانة بكائن Me فمثلاً عند الإشارة إلى أي نموذج في إجراء VBA المخزن في وحدة النموذج النمطية، يكون من الأفضل استخدام كائن Me لتجنب الوقوع في أخطاء، إذا ما حدث لسبب غير متوقع أن اصبح الكائن نشطاً بدلاً منّ النموذج.

كائن DoCmd

كائن DoCmd هو كائن غير عادي يتوافر فقط في VBA، وهو لا يمثل كياناً مادياً له خصائصه وطرقه، بل يمثل اتصالاً هاماً بين الماكرو وبين برمجة VBA. ومعظم العمليات التي تقوم بها بواسطة كائنات Application تزيد من الخطوات التي تتبعها عند العمل بطريقة تفاعلية في أكسس. وتعرف هذه العمليات بإجراء ماكرو. وتتوافر معظم إجراءات ماكرو في VBA كطرق لكائنات DoCmd.

كائن DoCmd والأتمتة

أخرج مايكروسوفت VBA كلغة مشتركة لجميع تطبيقات أوفيس، وتم الاستقرار على الأتمتة "تعرف باسم أتمتة OLE في الإصدارات السابقة" كطريقة لتطبيق واحد للعمل مع كائنات في تطبيق آخر. وبالنسبة للتطبيقات الخارجية مثل اكسل أو تطبيق Visual Basic الذي يأتي مفرداً، حتى تستطيع استخدام الأتمتة للعمل بكفاءة مع اكسس لابد من وجود طريقة لهذه التطبيقات الخارجية لاستبعاد إجراءات ماكرو. في نموذج الأتمتة يعمل أحد التطبيقات مثل أكسس. على سبيل المثال،على توافر الكائنات الخاصة به في تطبيقات أخرى، على الرغم من أن التطبيق الخارجي يعد تطبيقاً برمجياً، فإنه يمكن أن يطلب من كائن أكسس أن يستخدم طرقه. لتوفير إجراء ماكرو في تطبيق خارجي في الأتمتة، يجب أن يكون إجراء ماكرو طريقة لأحد الكائنات، لذا كانت أسهل الحلول هي إنشاء نموذج صناعي يمكن أن يتخذ من إجراءات ماكرو طرقاً له، ومن هنا، وجد كائن DoCmd.

يمكن لتطبيق خارجي أن يستبعد كائن ماكرو من خلال الأتمتة، وذلك بالطلب من كائن DoCmd استخدام الطريقة التي تناسب إجراء ماكرو. معظم إجراءات ماكرو في VBA تتوافر كطرق لكائن DoCmd.

مع الانتباه إلى متطلبات الأتمتة، كان على مايكروسوفت تقرير كيفية تعامل VBA ي أكسس مع العمليات التي تنفذها إجراءات ماكرو، ومعظم إجراءات ماكرو كان يمكن تعريفها كطرق لكل كائنات أكسس الأخرى التي تطبق عليها. فعلى سبيل المثال، إجراء ماكرو MoveSize الذي يستخدم لنقل وتغيير حجم النافذة النشطة كان يمكن تعريفه كطريقة لكائن Form و Report لا كطريقة لكائن DoCmd. وعن طريق القرارات التي اتخذت بشأن إجراءات ماكرو، يمكن القول بأنها غير متينة، فعلى الرغم من أن لإجراءات ماكرو طرقاً مناسبة لكائن DoCmd فإن العديد من إجراءات ماكرو قد تم تعريفها أيضاً كطرق لإجراءات محددة. فعلى سبيل المثال يتم تعريف Requery كطريقة لكائن Form وكائن Control وكائن DoCmd، ويتم تعريف Repaint كطريقة لكائن Form وكذلك يتم تعريف RepaintObject كطريقة لكائن DoCmd. ومن جانب آخر، يتم تعريف Close فقط كطريقة لكائن DoCmd وليس كطريقة لكائن مغلق.

كائن DoCmd وإجراءات ماكرو

يمكن استخدام كائن DoCmd لتشغيل ٤١ طريقة من إجراءات ماكرو التسع وأربعون كطرق VBA مناسبة. ويوضح "جدول ٥-١٧" إجراءات ماكرو الثمانية التي ليس لها طرق DoCmd مناسبة ويشرح كيفية تناول هذه العمليات في VBA في أكسس:

الجدول ٥-١٧: إجراءات ماكرو بدون مثيلاتها المناسبة من VBA

إجراء ماكرو	مثيلة من VBA في أكسس
Macro Action	لا يوجد لها مثيل.
AddMenu	استخدام وظيفة ( ) MsgBox.
MsgBox	استخدام وظيفة ( ) Shell.
RunApp	يقوم إجراء VBA باستدعاء إجراء آخر مستخدماً جملة Call، وكلمة Call الأساسية تكون اختيارية.
RunCode	استخدم جملة SendKeys.
SendKeys	استخدم جملة التعيين باستخدام علامة (=) كعامل تشغيل للتعيين.
SetValue	ليست هناك طريقة لإيقاف تشغيل إجراءات ماكرو في إجراء VBA، استخدم جملة Stop لتعليق استبعاد الإجراء أو جملة End لإنهاء الاستبعاد.
StopAllMacros	ليست هناك طريقة لإيقاف تشغيل إجراءات ماكرو في إجراء VBA، استخدم ExitSub أو ExitFunction أو ExitProcedurse للخروج من إجراء VBA.

ملاحظة لمزيد من المعلومات عنة إجراء ماكرو، انظر .MasterIng Access 2000 PremIum EdItIon by Alan Simpson and Celeste Robinson (Sybex, 1999).

طرق كائن DoCmd

بالنسبة لإجراءات ماكرو الحادية والأربعون التي لها طرق DoCmd مناسبة، تقوم تقو م الطريقة بتنفيذ المثيل لإجراء الماكرو المناسب بواسطة وسائط الإجراءات والتي تذكر كوســـــائط طـــرق بنفس الترتيب الذي تظهر به وسائط الإجراءات في نافذة ماكرو Design

ملاحظـــة تختلف بعض طرق كائن DoCmd عن إجراء ماكرو، تلك الاختلافات التي تذكر فــــي جــدول الاختلافـــات بيـــن DoCmd Methods، وthe Corresponding Macro Actions على القرص المضغوط. وفي بعــض الأحيان، على الرغم من أن طريقة DoCmd قد تناسب إجراء ماكرو، فأنـــه قد تكون هناك طريقة أفضل لتنفيذ العملية يوضح جدول "طـــرق DoCmd والتقنيات المفضل استخدامها"على القرص المضغوط مثـــل هـــذه الطـــرق، وستجد الجدولان السابقان في Tables\Chapter5.pdf.

لا تحيد طرق كائن DoCmd أي قيم أو كائنات، لذلك يكون بناء الجملة كالآتي:

```
DoCmd.method argument1, argument2, ..., argumentN
```

فعلى سبيل المثال، يكون لإجراء ماكرو GoToControl مع وسيطة اسم أداة التحكم طريقـــة GoToControl مماثلة تماماً ويكون بناء جملتها كالآتي:

```
DoCmd.GoToControl controlname
```

ControlName: هو، تعبير سلسلة أي هو أداة التحكم للنموذج النشط أو لورقة البيانات.

وتستخدم وسائط الطرق ثوابتاً حقيقية محل قوائم وسائط إجراءات ماكرو المضمنــــة. فعلـــى سبيل المثال، تستخدم وacTable, acQuery, acForm, acReport, acDataAccessPage, acMacro, acModuleمحل قائمة لكائنات Database. كما تستخدم وســـائط الطـــرق قيـــم boolean وTrue "-1" وFalse "0" بدلاً من وسيطة Yes وNo لوسيطة إجراء ماكرو.

وهناك اختلاف جوهري بين إجراء ماكرو وطريقته، وهذا الاختلاف هو أنـــه كلمـــا تحـــدد وسيطة لإجراء ماكرو اسم كائن، تكون وسيطة الطريقة تعبير سلسلة هو في حد ذاته اسم الكائن. فعلى سبيل المثال، يجب أن تكون وسيطة FormName لإجـــراء مـــاكرو Openform اســـماً لنموذج محدد، لكن وسيطة fprmname لطريقة OpenForm ما هي ألا تعبير قد يكون متغيراً.

يوجد نوعان لطرق DoCmd للعمل في كائن Access Application في النوع الأول يجـــب تحديد الكائن كوسيطة للطريقة، وفي النوع الثاني تطبق طريقة DoCmd على الكائن النشط، فـــلا

تحتاج إلى تحديد الكائن كوظيفة، وفي المثالين الأول والثاني التاليين، تفتح طرق OpenReport و OpenForm كائنات يجب تحديدها كوسائط، بينما تعمل الطرق مع الكائن النشط في الثلاث أمثلة الأخيرة.

البيان	الوصف
DoCmd.OpenReport "Customer labels", acPreview	عند استخدام طريقة Open Report حدد التقرير كوسيطة للتقرير
Strname= "customer' DoCmd. OpenForm strname	تقوم الجملة الأولى بإنشاء متغير strname وبتعيين القيمة "Customers" وتفتح الجملة الثانية نموذج Customers
DoCmd.GoToRecord , , acNext؛	تنقل هذه الجملة التركيز ليصبح على السجل التالي في النموذج النشط أو في ورقة البيانات.
DoCmd.GoToControl "Country"	تنقل هذه الجملة التركيز ليصبح على أداة التحكم.المعروفة باسم Country في النموذج النشط.
DoCmd.Close	تغلق هذه الجملة النافذة النشطة.

ملاحظة عندما تكون وسيطة الطريقة تعبير سلسلة، أي عندما تكون اسماً لكائنة يحتوي على مسافات لا تحيط الاسم بأقواس مربعة فعلى سبيل المثال، استخدام "[customer Labels]" كوسيطة Reportname ينتج عنه خطأ ما لذا استخدم "Customer Labels".

مجموعة الخصائص"Properties"

يحتوي كل كائن Access Application على مجموعة Properties ذات مجموعة من كائنات Prope المضمنة، التي عادة تعرف باسم الخصائص. ستتعرف في الفصل الرابع عشر على إنشاء خصائص مخصصة للنماذج والتقارير، التي لا تضاف إلى مجموعة Properties ص فقط للخصائص المضمنة.

ولمجموعــة Properties نفس خــاصيتي مجموعــات Forms وReports وmodules وcontrols وapplication وcount. لتتعرف على خاصية Count مع هذه المجموعــة، افتــح نموذج Customer في عرض Form وجرب هذه النماذج:

- اكتب Type ? Forms!Customers.Properties.Count فــي نــافذة Immediate واضغط Enter وستعرض خصائص النموذج المضمنة التي يبلغ عددها ١٣٢ في نــافذة Immediate.
- اكتــب Type ? Forms!Customers!Country.Properties.Count ? واضغــط Enter وستعرض خصائص أداة تحكم مربع التحرير والسرد المضمنة التي يبلغ عددهــا ٨٢ في نافذة Immediate.

كائن الخاصية "Property"

لكل خاصية مضمنة لكائن Access كائن Property مطابق فعلى سبيل المثال، خاصية Visible لأداة التحكم لها كائن Property مطابق ولكائن Property خصائصه والتي تتضمن الخصــائص المذكورة في "الجدول ٥-١٨".

الجدول ٥-١٨: خصائص كائن Property

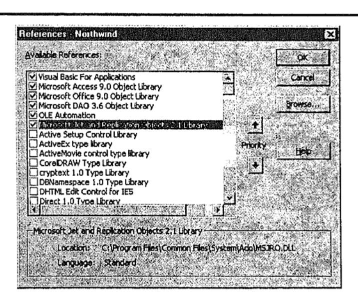
الخاصية	البيان
Name	سلسلة تحدد الخاصية.
Value	متغير يحتوي على إعداد الخاصية.

للتعرف على كيفية عمل خاصية Value، افتح نموذج Customers وجرب الأمثلة الآتية:

- Type ? Forms!Customers!Country.Properties!Visible.Value فــي نــافذة Immediate واضغط Enter ستعرض نافذة Immediate True.
- اكتب Type ? Forms!Customers!Country.Visible واضغــط Enter ســتحصل على نفس النتيجة نظراً لأن Properties هي المجموعــة الافتراضيــة لأداة التحكــم، وvalue هي الخاصية الافتراضية.

مجموعة المراجع "References"

عند إنشاء تطبيق Access يستخدم VBA لمعالجة البيانات أكسس أخرى، أو ي تطبيقات أخرى مثل مايكروسوفت اكسل وورد، يجب إعطاء معلومات عن الكائنات للتطبيق الآخر وذلك بإنشاء مرجع لمكتبة نوع التطبيق. وعادة تقوم بإنشاء وإزالة المراجع بطريقة تفاعلية باستخدام مربع حوار References "اختر Tools ⇦ References" وذلك عندما تكون نافذة Module هي النافذة النشطة "انظر الشكل ٥-٩" كما يمكن إضافة وإزالة مراجع بطريقة مبرمجة باستخدام VBA.



الشكل ٥-٩

استخدام مربع حوار References لإنشاء مرجع لمكتبة نوع قاعدة بيانات أكسس أخرى، أو لتطبيق آخر.

يمثل كل مرجع كائن Reference، كما تحتوي على مجموعة References لكائن application على كائنات Reference المعدة حالياً لقاعدة البيانات. ولمجموعة References خاصية Count التي يمكن استخدامها لتحديد أرقام المراجع المعدة حالياً، ولها أيضا الطرق الموضحة في "الجدول ٥-١٩". عند إضافة مرجع بطريقة مبرمجة تتعرف مجموعة References على حدث ItemAdded. وكذلك الحال عند إزالة مرجع بنفس الطريقة.

الجدول ٥-١٩: خصائص كائن Property

الطريقة	الشرح
CreateFromFile	تقوم بإعداد مرجع لمكتبة النوع باستخدام المسار إلى الملف، مثل ملف مكتبة النوع ملف قاعدة بيانات، ملف مستبعد أو أداة تحكم ActiveX.

الجدول ٥-١٩: خصائص كائن Property

الطريقة	الشرح
CreateFromGUID	تقوم بإعداد مرجع لمكتبة النوع باستخدام معرف النسخة المماثلة للملف "GUID".
Item	تعيد عنصراً محدداً من المجموعة فقط حدد فهرس كائن References أو اسمه وitem هو العنصر الافتراضي لمجموعة References.
Remove	تزيل مرجعاً من مجموعة References.

كائن المرجع "Reference"

يمثل كائن Reference مرجعاً محدداً معد حالياً لمكتبة النوع لتطبيق آخر أو لقاعدة بيانات أكسس أخرى. يمكن الإشارة إلى كائن Reference باستخدام بناء الجملة العادي لأي عنصر في مجموعة. فمثلاً للإشارة إلى مكتبة نوع أكسس استخدم بناء الجملة Access !References ولكائن Reference الخصائص المذكورة في "الجدول ٥-٢٠" لكن ليس له طرق.

الجدول ٥-٢٠: خصائص VBA الوحيدة لكائن Reference

الخاصية	التشغيل	قيمة الإرجاع	الشرح
BuiltIn	Read-only	Boolean	تحدد ما إذا كان كائن Reference يمثل مرجعاً افتراضياً لا يمكن إزالته عند تشغيل أكسس بطريقة صحيحة.
Collection	Read	OnlyObject	تشير إلى مجموعة References.
FullPath	Read-only	String	تحدد مسار واسم مكتبة النوع.

الجدول ٥-٢٠: خصائص VBA الوحيدة لكائن Reference

الخاصية	التشغيل	قيمة الإرجاع	الشرح
GUID	Read-only	String	تحدد معرف النسخة المماثلة لمكتبة النوع.
IsBroken	Read-only	Boolean	تحدد ما إذا كان كائن References يشير إلى مرجع صالح في Rejistry. استخدام هذه الخاصية لتحديد ما إذا كانت مكتبة النوع قد تم نقلها أو حذفها.
Kind	Read-only	لها القيمة ١ لمشروع VBA والقيمة صفر لمكتبة النوع.	تحدد ما إذا كان كائن References يمثل قاعدة بيانات أكسس أخرى أو مكتبة نوع.
Major	Read-only	Long	تحدد رقم الإصدار الأساسي لتطبيق يمثله كائن Reference.
Minor	Read-only	Long	تحدد رقم الإصدار الثانوي لتطبيق يمثله كائن Reference.
Name	Read-only	String	تحدد التعبير السلسلة التي يعرف كائن Reference.

كائنات أكسس VBA

لأكسس VBA ثلاث كائنات هي: Err و Debug و Collection.

كائن Err

يحتوي كائن Err على معلومات عن الخطأ الذي ينشأ في إجراء VBA أثناء تشغيل الإجراء "يعرف باسم خطأ VBA وقت التشغيل". تعبأ خصائص كائن Err بمولد الأخطاء مع الاستعانة بالمعلومات الوحيدة التي تعرف الخطأ."يمكن أن يتولد خطأVBA وقت التشغيل بواسطة كائن، أو VBA أو بواسطتك أنت كمبرمج VBA" ونظراً لأنة لا يوجد سوى كائن Err واحد، فإنك لن تتعامل سوى مع خطأVBA وقت التشغيل واحد أيضاً. تمحي خصائص Err "المعاد تشغيلها إلى صفر أو السلسلة ذات الطول صفر" تلقائياً بعد أي جملة Resume أو OnError أو ExitSub أو ExitFunction أو ExitProperty.

يوضح "الجدول ٥-٢١" خصائص كائن Err، بينما يوضح "الجدول ٥-٢٢" الطرق الخاصة به.

الجدول ٥-٢١: خصائص كائن Err

الخاصية	التشغيل	الشروح
Number	Read/write	رقم صحيح طويل يطابق خطأ محدداً في قائمة التعليمات البرمجية الصالحة لخطأ VBA وقت التشغيل.
Source	Read/write	اسم الكائن أو التطبيق الذي هو في الأصل نتج عنه الخطأ.
Description	Read/write	رسالة تطبق خطأ محدداً.
HelpFile	Read/write	المسار ذو الإمكانات الكاملة لملف Visual Basic Help حيث توجد تعليمات عن الخطأ المحدد.
HelpContext	Read/write	معرف السياق لملف Visual Basic Help الذي يعرف موضعاً ما في Help File الذي يقدم تعليمات عن الخطأ المحدد.
LastDLLError	Read/Only	التعليمات البرمجية للخطأ لآخر استدعاء لـ DLL

الجدول ٥-٢٢: طرق كائن Err

الطريقة	الشرح
Raise	ينتج عنها خطأ VBA وقت التشغيل. استخدم هذه الطريقة لمحاكاة خطأ VBA وقت التشغيل وذلك بتحديد وسيطة الطريقة Number كتعليمات برمجية صالحة لخطأ VBA وقت التشغيل. إضافة إلى ذلك، يمكن إنتاج أخطاء مخصصة يحددها المستخدم وذلك بتحديد وسيطة الطريقة Number كرقم صحيح Long لا يطابق التعليمات البرمجية الصالحة لخطأVBA وقت التشغيل.
Clear	تمحو كل إعدادات الخصائص لكائن Err بوضوح. استخدم هذه الطريقة عند تأجيل تناول الخطأ، وذلك باستخدام جملة On Error Resume Next statement.

كائنDebug

ليس لكائن Debug خصائص، لكن له طريقة واحدة هي Print التي تطبع النص في نافذة Immediate، ويكون بناء الجملة [outputlist] Debug Print عند العمل في نافذة Immediate، يمكن استخدام علامة الاستفهام كاختصار "[outputlist] ?" لكن عند طبع شيء من إجراء VBA في نافذة Immediate، بناء الجملة بالكامل للإشارة إلى كائن Debug.

كائن المجموعة "Collection

يشير كائن Collection إلى مجموعة عناصر من نفس النوع، والغرض من هذا الكائن إتاحة إمكانية إنشاء المجموعات الخاصة بك "ستستخدم كائن Collection في الفصل الرابع عشر لإنشاء كائنات نماذج عديدة، أي أمثلة عديدة لوحدة النموذج النمطية".

عند إضافة عناصر إلى كائن Collection، يتم فهرستها تلقائياً ولاحظ أن فهرس المجموعة التي يعرفها المستخدم تبدأ بالرقم واحد وليس بالرقم صفر. ولكائن Collection نفس خصائص Application وcount " لتحديد أرقام عناصر المجموعة"تماماً مثل كائنات المجموعة المختلفة، بالإضافة إلى الطرق المذكورة في "الجدول ٥-٢٣".

الجدول ٥-٢٣: طُرق كائن Collection

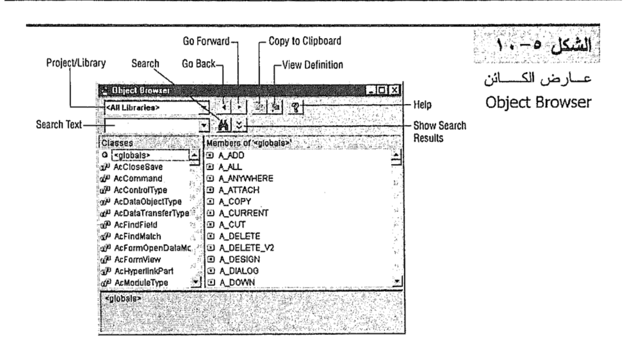
الطريقة	الشروح
Add	تضيف عنصراً إلى كائن Collection
Remove	تزيل عنصراً من كائن Collection
Item	تعيد عنصراً محدداً من عناصر كائن Collection

تلميح يستخدم مايكروسوفت أوفيس ٢٠٠٠ مجموعة من الميزات المشتركة، مثل أشرطة الأوامر لتخصيص أشرطة القوائم والقوائم المختصرة وأشرطة الأدوات، وأوفيس Assistant الذي يقدم نوعاً جديداً من التعليمات الفورية، وfileSearch الذي يقدم إمكانات محسنة للبحث عن ملف وتعمل عادة مع هذه الميزات بطريقة تفاعلية، غير أن أوفيس ٢٠٠٠ يقدم نماذجاً لكائنات بحيث يمكن التحكم في وتخصيص تلك الميزات بطريقة مبرمجة في VBA.

استخدام عارض الكائنات Object Browser

يعد استخدام Object Browser طريقة جيدة لمعرفة المزيد عن كائنات AccessApplication. وكائنات تشغيل البيانات، والكائنات التي توفرها تطبيقات أخرى من خلال الأتمتة، وذلك لأنه يعرض المعلومات في مكتبة النوع للتطبيق، التي تعد الملف الذي يتضمن معلومات عن طرق وخصائص كائنات التطبيق، وعن الثوابت الحقيقية التي يستخدمها التطبيق.

للوصول إلى Object Browser، اضغط Alt+F11 لفتح Visual Basic Editor، ثم حدد View ⇦ Object Browser. يوضح "الشكل ٥-١٠" نافذة Object Browser.

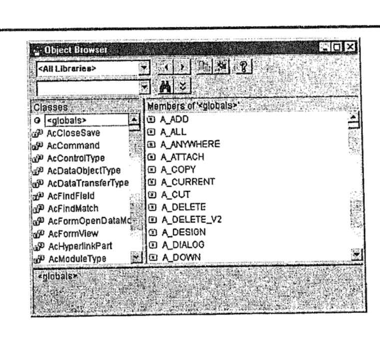


الشكل ٥-١٠

عارض الكائن Object Browser

يحتوي مربع التحرير والسرد والتحديد Project/Library في أعلى النافذة على قائمة بمكتبات النوع التي تعرفها قاعدة البيانات الحلية وبدل الاختيار الافتراضي <All Libraries> على أن المربع يتضمن عناصر من جميع مكتبات النوع، فقط حدد مكتبة النوع التي تريد العمل معها.

استخدم مربع التحرير والسرد Search Text للبحث عن عنصر ما في المكتبة المحددة، ثم ادخل سلسلة النص التي تريد البحث عنها وأنقر زر Search. يوضح "الشكل ٥-١١" نتيجة إيجاد سلسلة النص "line" انقر زر Show Search Results بالسهم المزدوج لإخفاء أو عرض نتائج البحث.



الشكل ٥-١١

يمكن البحث عن سلسلة نص في مكتبة نوع محددة

يحتوي مربع القائمة Classes على الجانب الأيسر على قائمة بمحتويات المكتبة المحددة، بما في ذلك الثوابت والكائنات وتبدأ مكتبة النوع المحددة بالعنصر <globals> ويرجع سبب تسمية هذا المربع بهذا الاسم، إلى تعريف الكائن الذي يسمى فئة الكائن. عند إنشاء كائن جديد، كنموذج على سبيل المثال، استخدم الوحدة النمطية الفئوية للنموذج كتعريف له، أو استخدم مخططاً لإنشاء النموذج كمثال الفئة.

عند تحديد عنصر في مربع القائمة Classes سيتغير مربع القائمة Members ليعرض معلومات عن العنصر مثل الثوابت والخصائص والطرق والأحداث. وعند تحديد عنصر في مربع القائمة Members، سيعرض اسفل نافذة Object Browser بعض المعلومات مثل بناء الجملة الخاص بالعنصر المحدد. انقر زر Help أو أضغط F1 لعرض تعليمات فوريه عن العنصر المحدد استخدم أزرار Go Back وgo Forward للانتقال بين العناصر المحددة.

إذا كان العنصر المحدد في مربع القائمة Classes وحدة نمطية قياسية أو فئوية، انقر زر View Definition لفتح وعرض الوحدة النمطية.في حالة تحديد إجراء في الوحدة النمطية ثم نقر زر View Definition سيتم فتح الوحدة النمطية وبها نقطة الإدراج في الإجراء المحدد. للصق تعليمات برمجية في وحدة نمطياً، اختر عنصراً في مربع القائمة Members، وانقر زر Copy to Clipboard، ثم انقر في الوحدة النمطية في المكان الذي تريد لصق العنصر به وأخيراً اضغط Ctrl+V للصق التحديد.

يمكن أيضاً استخدام Object Browser لاستكشاف " الوصول إلى" مكتبة النوع أكسس ومشروع أكسس وذلك كالآتي:

١- حدد أكسس في قائمة التحرير والسرد، سيتضمن مربع القائمة Classes في الجانب الأيسر، قائمة بكائنات Access Application، وفئات الثوابت الحقيقية.

٢- حدد فئة كائن Form في مربع القائمة Classes، وحدد Go To Page في مربع القائمة Members، الذي سيعرض خصائص وطرق وأحداث فئة كائن Formباستخدام أيقونات مختلفة لكل نوع من أنواع العناصر . سيعرض بأسفل نافذة Object Browser بناء الجملة الخاص بالطريقة "انظر الشكل ٥-١٢ أ"

٣- انقر زر Help لـ Object Browser لعرض تعليمات فورية عن عنصر المجموعة المحدد. عد مرة أخرى إلى نافذة Immediate.

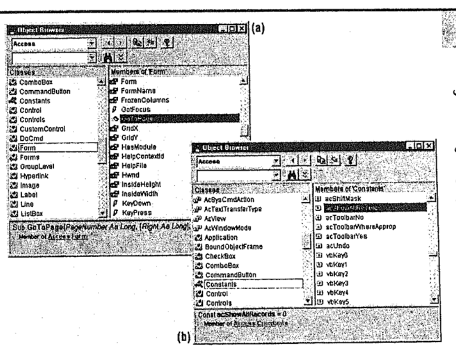
٤- انقر زر Copy to Clipboard في Object Browser لنسخ قالب تعليمات العنصر البرمجية. يمكن الانتقال إلى نافذة Module ولصق قالب التعليمات البرمجية في الوحدة النمطية.

٥- حدد Constants في مربع القائمة Classes، وسيعرض مربع القائمة Members الثوابت الحقيقية التي يقدمها كلاً من أكسس وVBA. عند تحديد أحد الثوابت، تعرض قيمته الرقمية بأسفل النافذة "انظر الشكل ٥-١٢ ب".

٦- اختر مشروع Northwind في مربع التحرير والسرد Project/Libraries. وبالافتراض لن يتغير اسم المشروع عند إنشاء نسخة جديدة لقاعدة البينات، فتذكر قاعدة البيانات الحالية باسم مشروعها. عند تحديد مشروع أكسس، يذكر مربع القائمة Classes كل الوحدات النمطية القياسية والفئوية التي قمت بإنشائها، ويذكرها جميعاً بالخط الأسود العريض. وعند تحديد وحدة نمطية في مربع القائمة Classes، يتغير مربع القائمة Members الموجود بالجانب الأيمن ليعرض معلومات عن هذه الوحدة النمطية.

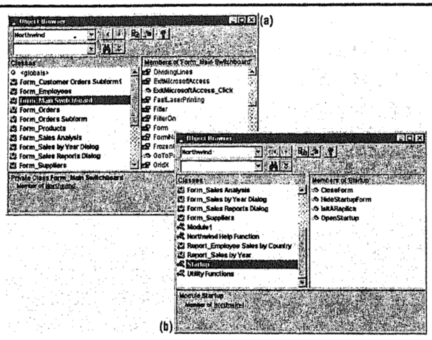
٧- حدد وحدة النموذج النمطية Form Main Switchboard في مربع القائمة Classes. يتضمن مربع القائمة Members خصائص النموذج المضمنة والأحداث والطرق كما يعرض، بالخط الأسود العريض، أسماء الإجراءات المخزنة في الوحدة النمطية مستخدماً أيقونات مختلفة لكل نوع من أنواع العناصر "انظر الشكل ٥-١٣ أ"

٨- اختر الوحدة النمطية القياسية Startup. عند اختيار وحدة نمطية قياسية أو وحدة نمطية فئوية مستقلة، يعرض مربع القائمة Members بالخط الأسود العريض أسماء الإجراءات المخزنة في الوحدة النمطية "انظر الشكل ٥-١٣ ب"



الشكل ٥-١٢

يستخدم Object Browser أيقونات مختلفة لعناصر فئة Form بما في ذلك الخصائص والطرق والأحداث، كما يعرض بناء الجملة الخاص بالعنصر المحدد "أ" اختر Constant لعرض تطبيق أكسس وثوابت VBA "ب".



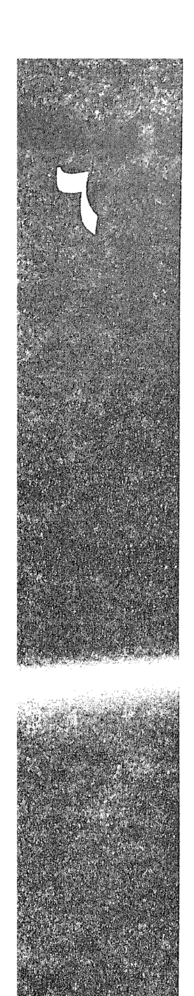
الشكل ٥-١٣ استخدم Object Browser لاستكشاف مشروع أكسس. يمكن عرض الوحدة الفئة النمطية لنموذج أو تقرير "أ"، أو عرض الإجراءات في وحدة نمطية قياسية أو وحدة نمطية فئوية مستقلة "ب"

خلاصة

اتسعت جولة كائن Access Application في هذا الفصل لتغطي أهم ميزات برمجة VBA ولقد كانت النقاط الهامة في هذا الفصل كالآتي:

- الكائن هو عنصر، تحدد هويته بمجموعة خصائص التي تحدد بدورها خصائص الكائن، وبمجموعة طرق تحدد الإجراءات التي يمكن للكائن القيام بها.والكائنات في نموذج Access Application لها خصائص وطرق تتوافر فقط في برمجة VBA، كما أن هناك بعض الكائنات في نموذج Access Application لا تتوافر سوى في برمجة VBA ومن هذه الكائنات كائن Module ومجموعة Modules وكائن Reference ومجموعة References.
- يمكن إعداد خاصية لقيمة النص باستخدام جملة تعيين كالآتي:Object Propertyname = Value. يمكن قراءة أو تهيئة قيمة نص لخاصية أو تعيين القيمة لمتغير باستخدام جملة تعيين كالآتي: Variable = Propertyname إذا كانت الخاصية تعيد كائناً، قم بتهيئة مرجع الكائن وتعيين المرجع لمتغير كائن باستخدام جملة تعيين كالآتي: setObjectVariable = Objectpropertyname.

- يمكن تشغيل أو استدعاء الطريقة باستخدام بناء الجملة : Objectmethod. ومعظم الطرق لا تعيد شيئاً،لكن في بعض الأحيان قد تعيد قيمة نص أو كائن أ. كما أن لمعظم الطرق معلومات إضافية يجب تحديدها كوسائط طرق قبل تشغيل الطريقة.
- يمكن معالجة الكائن إما بإعداد خاصية، أو بتشغيل إحدى طرق الكائن أو بتشغيل طريقة كائن DoCmd أو بتشغيل وظيفة أو جملة أكسس مضمنة. ويستخدم كائن DoCmd لتشغيل طرق مماثلة لمعظم إجراءات ماكرو.
- يمكن استخدام خاصية Recordsetclone للنموذج لإنشاء مؤشر سجل حالي مستقل في مجموعة سجلات النموذج، ويمكن استخدام مؤشر السجل الحالي للنسخة للتجول بمجموعة السجلات دون المساس بالمؤشر الذي يعرض في النموذج.
- يمكن استخدام خاصية Bookmark للنموذج لتخزين الإشارة المرجعية للسجل الحالي في متغير ثم الرجوع إلى سجل معلم فيما بعد.
- يمكن استخدام خاصية Me للنموذج أو للتقرير للإشارة إلى النموذج أو التقرير في إجراء ما، مخزن في وحدة النموذج أو التقرير النمطية ويعد استخدام مرجع Me أسرع وسيلة للإشارة للنموذج أو التقرير.



فهم DAO وأنواع كائن ADO

يقدم هذا الفصل محرك قاعدة البيانات في مايكروسوفت أكسس، مثل تلك في كائنات برامج VBA. مع VBA، يمكن التحكم في الوقت وكيفية تبادل التطبيق مع المحرك حيث يؤدي المحرك إلى الوصول للكائنات التي من خلالها يتم تحديد عملية التبادل. والكائنات التي يجعلها المحرك متاحة هي كائنات الوصول إلى البيانات وهي جزء من نوع "DAO "Data Access Object. وتوجد لدى كل كائن خواصه وطرقه الخاصة به. وفي هذا الفصل، هناك بعض التدريبات على كائنات الوصول إلى البيانات والخواص والطرق التي يحتاج المستخدم إليها عند تعلم استخدام VBA لجعل قاعدة بيانات أكسس آلية.

كما يقدم هذا الفصل الكائنات التي تُعد جزءاً من نوع ADO "ActiveX Data Objects". وبالنسبة لمشاريع أكسس، يقوم MSDE بإدارة الكائنات في شكل ADO الهيكلي.

خدمات إدارة قاعدة بيانات أكسس

يُعد محرك قاعدة البيانات Jet هو نظام إدارة لقاعدة البيانات في أكسس. حيث أن MSDE هو نظام الإدارة لقاعدة بيانات المشاريع في أكسس. فعند العمل بصورة تبادلية مع أكسس أو عند إنشاء وحدات ماكرو بجعل قاعدة البيانات آلية، فليس هناك ضرورة في التركيز على وظائف وإمكانات المكونين الأساسيين لأكسس. وهما طبقة التطبيق ومحرك قاعدة البيانات. حيث تقوم واجهة أكسس بهذا وخاصة ما يرتبط بمحرك قاعدة البيانات حيث يوجد لدى مايكروسوفت أكسس جميع الإرشادات الضرورية مثبتة في التعليمات الداخلية له. وفيما يلي توجد بعض الأمثلة لخدمات إدارة قاعدة بيانات أكسس المتاحة.

عند تصميم جدول أو استعلام في قاعدة البيانات، فإن طبقة التطبيق تستخدم الطرق المثبتة ذات التعليمات البرمجية لترتيب Jet ذلك لإنشاء وتخزين الكائن المصمم في أسلوب عرض Design.

- عند تصميم جدول لمشروع ما، فإن طبقة التطبيق مُصاحبة مع MSDE، تستخدم OLE DB للاتصال مع الخادم لإنشاء وتخزين الكائن المصمم في أسلوب عرض Design.
- عند تحديد خيارات مرجعية في إطار Relationships "يتم فتحه من Tools ⇦ Relationships " فسوف يدفع المحرك هذه الخيارات.
- عند الارتباط بجدول خارجي "عند اختيار File ⇦ Get External Data" فسوف تقوم الواجهة بالترتيب لمحرك قاعدة البيانات لإنشاء وإدارة الارتباط.
- عند تشغيل استعلام ما في قاعدة البيانات، فسوف ترسل الواجهة عبارة SQL المعادلة إلى Jet لبدء العملية: تقوم Jet بإنشاء مجموعة التسجيلات المناسبة وإرجاعها إلى أكسس لعرض بعض كائنات الواجهة.

- عند فتح أسلوب عرض في مشروع أكسس، فإن MSDE يرسل استعلامات عن طريق OLE DB إلى الخادم لتشغيل جدول تقديري مع البيانات من قاعدة البيانات المُخزنة في الخادم المضيف.
- عند تأمين قاعدة بيانات أكسس باستخدام واحد من مربعات حوار Security "يتم فتحه باختيار Tools ⇦ Security"، فسوف تقوم الواجهة بالترتيب في Jet لإنشاء وتخزين معلومات التأمين.
- عند محاولة مستخدمين أو أكثر تحرير نفس التسجيل أو عند محاولة مستخدم واحد العمل في إصدارين لنفس التسجيل، فإن تطبيق أكسس يقوم بالترتيب مع محرك قاعدة البيانات لمعالجة المشاكل الحادثة.

ملاحظة يتم الإشارة إلى تأمين مشاريع أكسس في نهاية الخادم. لمزيد من المعلومات عن تأمين خادم SQL، راجع "SQL Server 7 in Record Time" لمايكل جوندرلوي وماري شيبمان "سايبكس ١٩٩٩".

هناك ميزة خاصة في برمجة VBA وهي عمل الإجراءات لخدمات قاعدة البيانات هذه فمع تطبيق VBA، يمكن كتابة الإجراءات لإنشاء جداول واستعلامات جديدة وكذلك إنشاء كائنات مخصصة. كما يمكن تعديل الكائنات المثبتة عبر إنشاء خواص وطرق مخصصة لها. يمكن عمل الإجراءات التي تربط قاعدة البيانات بالجداول الخارجية والتي تقوم بتحديث الارتباط تلقائياً عند تحريك الجدول. وكذلك يمكن عمل الإجراءات لعمل المرجع وتأمين الشاشة وتعديل تأمين التسجيل.

ملاحظة باستخدام قاعدة بيانات مايكروسوفت أكسس، قد يتم تخزين البيانات في ملف mdb.. وعلى العكس، لا تحتوي فقط على كائنات الواجهة "مثل الأشكال والتقارير" وكائنات البرمجة "ماكرو والوحدات النمطية". يتم تخزين البيانات الأساسية في ملف قاعدة البيانات "مثل ملف dat. في SQL Server" على الخادم المضيف. أما ADO "ActiveX Data Objects" والتي سيتم تغطيتها لاحقاً في هذا الفصل فتحتوي على الطُرق القادرة على إرجاع البيانات سريعاً مع مختلف أنوع قاعدة البيانات.

محرك قواعد البيانات Jet وآلياته

يزودنا محرك قاعدة بيانات Jet بمجموعة من خدمات إدارة قاعدة البيانات لأي تطبيق يقوم بعمل الاتصال معه. ويُعد Jet تطبيق مستقل بذاته.

محرك قاعدة بيانات Jet هو خادم Com "كانت آلية OLE فيما مضى". وكان OLE يشير إلى ارتباط وتضمين الكائن كما كان مجموعة من القواعد التي تتبعها التطبيقات إذا كان هناك اشتراك في استخدام المستندات. أما OLE المسماة الآن COM فقد دخلت في تكنولوجيا أوسع تضم الآن معينات للميزات الإضافية متضمنة الآلية.

فالآلية هي تكنولوجيا تسمح للتطبيقات المسماة خادم COM بأن تنشر برامج عمل المسماة تعريف الفئة، مجموعة معينة من الكائنات. بهذا يمكن للتطبيقات الأخرى المسماة عناصر التحكم COM بإنشاء كائنات في تطبيق الخادم طبقاً لبرامج العمل هذه وتسمى الكائنات المنشئة كائنات COM.

يتم إنشاء البرامج في عنصر تحكم COM لتنفيذ العمليات القياسية للكائنات. فيمكن لعنصر تحكم COM عمل خواص كائنات COM وكذلك إرجاع طرق الكائنات. وتعد مايكروسوفت إكسل وProject وأكسس أمثلة لعناصر تحكم COM التي تستخدم الآلية للاستفادة من محرك قاعدة بيانات Jet. هذه التطبيقات الثلاثة هي أيضاً خادم COM والذي يتيح كائناتها الخاصة لأي تطبيق آخر.

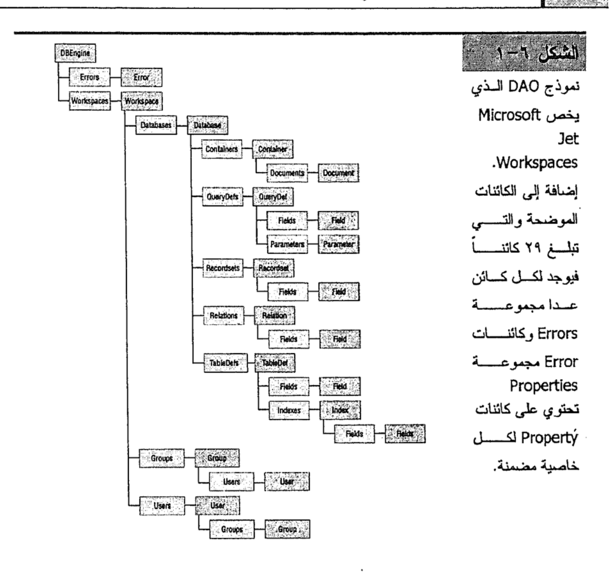
وكل من إكسل وبروجكت وأكسس برامج تنفيذية "exe". والبرامج التنفيذية هو نظام تشغيل يمكن أن ينفذ مجموعة خاصة من الوظائف. وهو يعمل بصورة عملية. فإذا كان خادم COM تنفيذي فيجب على عنصر تحكم COM أن ينشئ من عمليته الخاصة داخل عملية خادم COM. ويسمى خادم COM التنفيذي خادم COM خارج العملية. فمثلاً يمكن استخدام الآلية في أكسس لإنشاء كائن جدول بيانات في إكسل ثم التحكم فيها. ويُعد إكسل خادم COM خارج العملية. ولهذا، يجب على أكسس أن يخرج من عمليته الخاصة ليدخل في عملية إكسل لإنشاء جدول البيانات والتحكم فيه. يمكن أيضاً استخدام الآلية في أكسس للتحكم في نموذج آخر في أكسس والذي يعمل على أنه خادم COM خارج العملية. "عند بدء تشغيل نموذج آخر لبرنامج تنفيذي فإن النموذج الثاني يعمل في العملية الخاصة به: فيمكن مراقبة رمز كل نموذج في شريط مهام ويندوز".

معظم محتويات Jet تعد DLLs "مكتبات ارتباط حيوي" بدلاً من كونها تنفيذية. على الأخص آلية Jet المعتمدة على مجموعة خاصة من DAO DLLs. وعلى غير التنفيذيين فإن DLL يعمل بنفس طريقة التطبيق الذي استخدمه. يسمى خادم COM والذي يعمل على أنه DAO DLLs خادم COM داخل العملية. حيث يقوم DAO DLLs بالعمل في نفس العملية على أنه تطبيق العميل أسرع بكثير من خادم خارج العملية. ويعد Jet خادم COM داخل العملية.

عند استخدام التطبيقات مثل إكسل أو بروجكت Jet على أنه خادم COM، فإنها تستخدم تعليمات VBA لاستدعاء DAO DLLs التي تحتوي على الملفات التي تدخل أكسس إلى كائنات الوصول إلى البيانات. أما أكسس فلديه استدعاءات تخص Jet المكتوب في تعليماتها الداخلية. فعند العمل مع أكسس بصورة تفاعلية أو عند تشغيل وحدات ماكرو، فإن أكسس يقوم باستدعاء Jet مباشرة باستخدام التعليمات الداخلية. فعند استخدام تعليمات VBA في إكسس للعمل مع Jet فإن التعليمات تستدعي DAO DLLs.

شكل كائن الوصول إلى البيانات DAO الهيكلي

يوجد هيكل نموذج كائن DAO بصورة منفصلة من هيكل تطبيق أكسس وهيكل كائن أي تطبيق خارجي يتم استخدامه مع Jet. يوضح جدول ١-٦ ٣١ كائناً للوصول إلى البيانات كما يوضح شكل ١-٦ هيكل ٢٩ كائناً. "الكائنات غير الموضحة في الشكل هي مجموعة Properties وكائنات Property." يوجد في أعلى الشكل كائن DBEngine والذي يتم استخدامه للرجوع إلى وسيلة قاعدة بيانات Jet في إجراء VBA. أما الكائنات الأخرى فهي إما مجموعات أو أعضاء المجموعة إضافة لهذا، يوجد بكل كائن "عدا مجموعة Errors وكائنات Error" مجموعته الخاصة من Properties والتي تحتوي على كائنات Property المنفصلة لكل خاصية مضمنة.



الشكل ٦-١

نموذج DAO الذي يخص Microsoft Jet .Workspaces إضافة إلى الكائنات الموضحة والتي تبلغ ٢٩ كائناً فيوجد لكل كائن عدا مجموعة Errors وكائنات Error مجموعة Properties تحتوي على كائنات Property لكل خاصية مضمنة.

الجدول ٦-١: كائنات الوصول إلى البيانات لنموذج DAO

الكائن	الوصف
DBEngine	يمثل محرك قاعدة بيانات Jet.
Workspace	يمثل جلسة عمل Jet "تبدأ جلسة العمل عندما يبدأ المستخدم وتنتهي عندما ينتهي".
Workspaces	تحتوي على كل كائنات Workspaceالمعرفة بمحرك قاعدة بيانات Jet الحالي.

الجدول ٦-١ : كائنات الوصول إلى البيانات لنموذج DAO

الكائن	الوصف
Database	يمثل قاعدة بيانات مفتوحة حالياً. يمكن لقاعدة البيانات أن تكون قاعدة بيانات أكسس "ملف mdb." أو مصدر خارجي للبيانات "مصدر بيانات ODBC".
Databases	تحتوي على كل كائنات Database في Workspace.
TableDef	يمثل جدولاً محفوظاً في قاعدة بيانات. قد يكون الجدول محلي في قاعدة البيانات الحالية أو جدول مرتبط في قاعدة بيانات خارجية. يعرف كائن TableDef الجدول ولا يمثل البيانات المخزنة في الجدول.
TableDefs	يحتوي على كل الكائنات في قاعدة البيانات.
QueryDef	يمثل استعلاماً محفوظاً في قاعدة بيانات. يعرف كائن QueryDef الاستعلام متضمناً عبارة SQL الخاصة به ولا يمثل البيانات.
QueryDefs	يحتوي على كائنات QueryDef في قاعدة بيانات.
Field	يمثل حقل معين في كائن TableDef أو QueryDef أو Index أو Recordset. وتعرف كائنات Field لكل من كائنات TableDef وQueryDef وIndex وRelation حقول هذه الكائنات ولكن لا تحتوي على أي بيانات. أما كائنات Field لكائن Recordset فتعرف الحقول على أنها تحتوي على البيانات في خاصية Value للحقل.
Fields	يحتوي على كائنات Field في الجدول أو الاستعلام أو الفهرس أو

الجدول ٦-١: كائنات الوصول إلى البيانات لنموذج DAO

الكائن	الوصف
	العلاقة أو مجموعة السجلات.
Index	يمثل فهرس كحقل واحد أو عدة حقول لتحديد التسجيلات أو للبحث عن العمليات وفرزها.
Indexes	يحتوي على كل كائنات الفهرس في الجدول.
Relation	يمثل علاقة كحقل واحد أو عدة حقول في جدولين يربطان التسجيلات في الجدول.
Relations	يحتوي على كل كائنات Relation في الجدول.
Parameter	يمثل المعامل "قيمة غير معلومة" والتي يزوده المستخدم إلى استعلام المعامل قبل تشغيل الاستعلام.
Parameters	يحتوي على كل كائنات Parameter في الاستعلام.
Recordset	يمثل في الذاكرة مجموعة تسجيلات في الجدول أو تسجيلات ناتجة عن استعلام أو عبارة SQL يتم تشغيلها. يتم استخدام مجموعة السجلات لإرجاع وإضافة وتحرير وحذف التسجيلات في قاعدة البيانات. يجب فتح كائن Recordset عبر التعليمات البرمجية ويكون موجود فقط خلال تشغيل التعليمات البرمجية.
Recordsets	يحتوي على كل كائنات Recordset المفتوحة في قاعدة البيانات الحالية.
Property	يمثل خاصية لكائن البيانات.
Properties	يحتوي على كل كائنات Property لكائن معين في البيانات.
Container	يمثل كائناً عام يقوم بتخزين المعلومات الإدارية عن عنصر كائنا مخزونة تم إنشائها في تطبيق خارجي. في أكسس، تضم العناصر جداول واستعلامات "في نفس كائن Container" وكذلك العلاقات المخزونة والأشكال والتقارير وصفحات البيانات ووحدات ماكرو والنماذج وقواعد البيانات.
Containers	يحتوي على كائنات Container في قاعدة البيانات.

الجدول ٦-١ : كائنات الوصول إلى البيانات لنموذج DAO

الكائن	الوصف
Document	يمثل كائن معين مخزون تم إنشاؤه في تطبيق خارجي. يقوم كائن Document بتخزين التفاصيل الإدارية للكائن. وفي أكسس تضم الكائنات المخزونة كل جدول واستعلام وعلاقة مخزونة والشكل والتقرير وصفحة البيانات ووحدة ماكرو والنموذج وقاعدة البيانات.
Documents	يحتوي كائنات Document لقاعدة البيانات
Error	يمثل الخطأ الحادث خلال عملية لها كائن بيانات أو واجهة أكسس. قد يوجد لعملية واحدة عدة كائنات Error.
Errors	يحتوي على كائنات Error لعملية واحدة. وإذا أحدثت العملية التالية خطأ فإن مجوعة Errors تمحي وتستبدل بكائنات Error جديدة.
User	يمثل حساب المستخدم ويخزن المعلومات عن مستخدم واحد.
Users	يحتوي على كائنات User في مساحة العمل
Group	يمثل مجموعة من المستخدمين
Groups	يحتوي على كائنات Group في مساحة العمل

تلميح يمكن استخدام عارض الكائن Object Browser "الذي سبق ذكره في الفصل الخامس" لمعرفة المزيد عن كائنات بتشغيل البيانات اضغط Alt+F11 لفتح Visual Basic Editor ثم حدد View ⇦ Object Browser، وفي Object Browser حدد DAO في مربع التحرير والسرد Project/Libraries، ثم اختر الكائن الذي تريد التعرف عليه في مربع القائمة Classes وأخيراً اختر الخاصية أو الطريقة في مربع القائمة Members.

أنواع كائنات DAO

يمكن تمييز كائنات DAO في فئتين: فئة تضم الكائنات التي تحتفظ بملف ما "دائمة"، فيما عدا ذلك من كائنات "غير دائمة يكون ضمن الفئة الثانية".

تحفظ الكائنات الدائمة في ملف قاعدة البيانات ".mdb" أو في ملف قاعدة بيانات معلومات مجموعة العمل ".mdb" وتضم هذه الكائنات Database وtableDef وindex وqueryDef وparameter وrelation وuser وgroup وField لكائن TableDef وQueryDef وبالنسبة للكائن الدائم تكون كائنات Property الخاصة به دائمة أيضاً على عكس كائنات المجموعة "Collection" التي تحتوي عليها الكائنات الدائمة، فكائنات المجموعة يتم إنشاؤها من جديد عند فتح قاعدة البيانات كل مرة، فهي تتسم بالحيوية حتى أنها تتغير في كل مرة يتم فيها إضافة أو حذف عنصر من المجموعة كذلك الأمر بالنسبة بكائنات Container وDocument، فهي لا تعد كائنات دائمة على الرغم من أنها تحتوي على معلومات تنفيذية "إدارية" خاصة بالكائنات الدائمة. وتتضمن كائنات Container وDocument معلومات عن كائنات DAO والتي تمثل قاعدة البيانات، الجداول، الاستعلامات، والعلاقات، إلى جانب معلومات عن الكائنات التي تم إنشاؤها في أكسس والتي تمثل النماذج والتقارير وصفحات تشغيل البيانات والماكرو والوحدات النمطية.

وعن الكائنات غير الدائمة، نقول أنها لا تحتفظ في ملفات وهي تتضمن كائنات DBEngine وerror وworkspace وcontainer وdocument وrecordset وField لكائن Recordset وتكون كائنات Property للكائن غير الدائم غير دائمة أيضا، كذلك الأمر بالنسبة لجميع كائنات المجموعة "collection"، لذا يجب إنشاء كائن غير دائم كعنصر في مجموعة، قبل الإشارة إلى الكائن. فعلى سبيل المثال، يمكن إنشاء كائن Recordset جديد باستخدام طريقة OpenRecordset لكائن Database قبل العمل مع مجموعة السجلات.

أنواع خصائص DAO

يمتلك Jet نوعين من الخصائص. خصائص مضمنة وخصائص المستخدم المعرفة. والخصائص المضمنة هي التي يقوم Jet تلقائياً بإنشائها والمحافظة عليها. فعند إنشاء كائن بتشغيل بيانات جديد من خلال DAO مثل كائن QueryDef جديد، يقوم Jet تلقائياً بإنشاء مجموعة خصائص للاستعلام الجديد ويتضمن Jet فقط الخصائص المضمنة في مجموعة Properties الجديدة للكائن، هذه الخصائص التي تعرف الخصائص الأساسية للكائن.

أما عن خصائص المستخدم المعرفة لكائنات تشغيل البيانات فلها نوعان: إما أن تكون خصائص تقوم أنت بإنشائها بنفسك وأن تكون خصائص يقوم تطبيق مثل أكسس بإنشائها إضافة خاصية المستخدم المعرفة وذلك بإضافتها إلى مجموعة Properties للكائن. فعلى سبيل المثال،

إذا أردت وصف العرض من استعلام جديد، فيمكنك استخدام إجراء VBA لإنشاء خاصية مستخدم معرفة تسمى Purpose تضاف إلى مجموعة Properties الخاصة بالاستعلام.

عند استخدام أكسس لإنشاء كائن Jet، فإنه يقوم بإضافة خصائص عديدة تعرف باسم خصائص التطبيق المعرف "application-defined properties" فعلى سبيل المثال، عند إنشاء استعلام جديد في نافذة الاستعلام Design، تكون معظم خصائص الاستعلام المذكورة في ورقة خاصية الاستعلام خصائص تطبيق معرف وليست خصائص DAO مضمنة، وتشمل خصائص Description و OutputAllFields وعند الحديث عن Jet، يمكن القول بأن خصائص التطبيق المعرف لن تتوفر إلا بعد كتابة قيم لها في ورقة الخاصية المناسبة، أو بعد كتابة إجراءات لإضافة خصائص التطبيق المعرف إلى مجموعة الكائن Properties في حالة إدخال وصفاً في مربع الخواص Description بورقة خواص الاستعلام الخاصة بالاستعلام الجديد، يتم إضافة خاصية property لمجموعة Properties لهذا الاستعلام، أما في حالة عدم إعداد قيمة الخاصية في الوجهة، ثم أردت إعداد الخاصية في إجراء VBA، فإنه يجب عليك أولاً إنشاء الخاصية ثم إضافتها إلى مجموعة Properties للكائن في الإجراء.

ملاحظة

لا تنسى أن لكل كائن مجموعة Properties الخاصة به، لذا فعند إضافة خاصية Description لأحد الاستعلامات، لا قيم إضافة الخاصية نفسها إلى استعلام آخر. وإذا حاولت استرجاع خاصية property في استعلام لم تكتب به. أي وصف، يصدر Jet خطأ ما ويوضح الفصل الرابع عشر كيفية إنشاء خصائص المستخدم المعرفة وإضافتها.

تعد خصائص بدء التشغيل مثالاً رائعاً لخصائص التطبيق المعرف فهي خصائص كائن Database التي يعرفها أكسس، ويمكن إعداد معظمها في مربع Startup "يعرض بعد تحديد Tools ⇐ Startup". وهذه الخصائص لبدء التشغيل التي تقوم بإعدادها في مربع حوار Startup يتم إضافتها إلى مجموعة Properties لكائن Database، ولاستخدام إجراء VBA لإعداد خاصية بدء تشغيل لم تقم بإعدادها من قبل في مربع حوار Startup، يجب أولاً إنشاء الخاصية ثم إضافتها إلى مجموعة Properties في إجراء VBA.

وخاصية بدء التشغيل التي لا تتوافر في مربع حوار Startup، هي خاصية AllowBypassKey الهامة، التي تستخدم لتحديد ما إذا كان مفتاح Shift متاحاً عن طريق تمرير خصائص بدء التشغيل والماكرو AutoExec. ونظراً لأنه يمكن إعداد خصائص بدء التشغيل متعددة لحماية التطبيق عن طريق مثلاً إخفاء نافذة database فإن تعطيل مفتاح Shift كمفتاح

جانبي لبدء التشغيل، يعد تقنية هامة لحماية قاعدة البيانات. وسيوضح الفصل ١٤ كيفية تعطيل مفتاح Shift لمنع الدوران الجانبي لخصائص بدء التشغيل وماكرو AutoExec.

مراجع كائنات تشغيل البيانات

في إجراء VBA، عادة ما تعمل مع كائنات تشغيل البيانات مستخدماً عمليتين أساسيتين هما: قراءة الخواص وتغييرها، والطلب من الكائن استخدام إحدى طرقه لمعالجة نفسه. لكن قبل تنفيذ أياً من العمليتين، يجب عليك أولاً الإشارة إلى الكائن.

عند تشغيل أكسس، بتشغيل Jet تلقائياً الذي يقوم بعد ذلك بإنشاء كائن DBEngine جديد في الذاكرة ليمثل نفسه. ويعد هذا الكائن كائناً مؤقتاً يظهر فقط عند فتح أكسس يختفي عند إغلاقه كما يقوم Jet أيضاً تلقائياً بإنشاء كائن Workspace افتراضي ككائن مؤقت "لكن لتسهيل الأمر، سنتطرق فقط إلى الحالة التي لا يتوفر لها الأمن" عند فتح قاعدة بيانات يقوم Jet بإنشاء كائن Database في الذاكرة يمثل قاعدة البيانات التي تم فتحها.

وللإشارة إلى كائن تشغيل بيانات في إجراء VBA، ابدأ بالإشارة إلى كائن DBEngine. ثم مُر عبر المسار الهرمي الذي ينتهي بك إلى الكائن مسجلاً مراجع الكائنات وكائنات المجموعة التي مررت بها استخدام عامل التشغيل النقطة (.) عند الانتقال من كائن إلى إحدى مجموعاته، لكن للانتقال من كائن إلى إحدى مجموعاته، لكن للانتقال من مجموعة إلى أحد عناصرها، فإنه يمكنك استخدام أي من المراجع الأربعة التالية:

- استخدام عامل التشغيل نقطة التعجب (!) للإشارة بوضوح إلى العنصر باسمه
- استخدم بناء الجملة القوسي للإشارة إلى اسم العنصر
- استخدم بناء الجملة القوسي للإشارة إلى متغير العنصر
- استخدم بناء الجملة القوسي للإشارة إلى عنصر حسب وضعه بالمجموعة

جميع المجموعات في Jet بندأ من الرقم صفر، لكن عادة ما تستخدم أكثر من نوع من أنواع بناء الجملة، فعلى سبيل المثال، عند تشغيل أكسس، يتم تشغيل Jet تلقائياً الذي يقوم بعد ذلك بفتح كائن workspace افتراضي والذي سيشير إليه حسب موضعه باستخدام Workspaces(0). ويمكن الإشارة إلى قاعدة البيانات التي تفتحها بالاسم، لكن نظراً لأنها أول قاعدة بيانات تقوم بفتحها فيمكن أيضاً الإشارة إليها حسب موضعها مستخدماً Databases(0)

```
DBEngine.Workspaces(0).Databases(0)
```

وإذا أردت مثلاً الإشارة إلى خاصية ValidationRule لحقل CustomerID في جدول Customers في قاعدة البيانات الحالية، استمر في عبور المسار كالآتي:

```
DBEngine.Workspaces(0).Databases(0).TableDefs!Customers.Fields! _
CustomerID.Properties!ValidationRule
```

استخدام مجموعات افتراضية

من حسن الحظ أن لجميع كائنات تشغيل البيانات مجموعات افتراضية، فيمكنك دائماً اختصار المراجع بحذف أسماء المجموعات الافتراضية والتي تكون كالآتي:

الكائن	المجموعة الافتراضية
Workspaces	DBEngine
Databases	Workspace
TableDefs	Database
Fields	TableDef
Fields	Recordset
Parameters	QueryDef
Fields	Index
Fields	Relation
Documents	Container
Groups	User
Users	Group

ملاحظة تكون مجموعة Properties هي المجموعة الافتراضية للكائن عند الإشارة إلى خاصية مضمنة غير أنه عند الإشارة إلى خاصية مخصصة، فإنه يجب عليك تضمين مرجع بمجموعة Properties كالآتي: .object.Properties!customproperty

أفضل بين مرجع الكائن والخاصية المهمة باستخدام عامل التشغيل النقطة: object.property وباستخدام هذه المجموعان الافتراضية يمكن اختصار مرجع قاعدة البيانات الحالية ليكون DBEngine(0)(0)، ومن ثم استخدام بناء الجملة التالي لمرجع خاصية .ValidationRule

DBEngine(0)(0)!tblCustomers!CustomerID.ValidationRule

استخدام وظيفة CurrentDB

إذا كنت تستخدم تطبيقاً آخر، مثل إكسل. على سبيل المثال، للعمل مع قاعدة بيانات أكسس مستخدماً الأتمتة (Automation) يكون بناء الجملة الأكثر اختصاراً الذي يستخدم للإشارة إلى قاعدة البيانات كالآتي: DBEngine(0)(0). لكن عند استخدام أكسس للعمل مع قاعدة البيانات فإنه تكون هناك طريقة بديلة للإشارة إلى قاعدة البيانات الحالية، فإذا كنت تعمل في أكسس فإنه يمكنك استخدام إما بناء جملة Jet، DBEngine(0)(0)، أو بناء جملة أكسس، CurrentDB. وذلك للإشارة إلى قاعدة البيانات الحاليةز ومع الاختيار الثاني يكون بناء الجملة للنموذج كالآتي:

CurrentDB!Customers!CustomerID.ValidationRule

على الرغم من أن بناء جملة Jet وبناء جملة أكسس يشيران إلى نفس قاعدة البيانات، فإنهما يختلفان عن بعضهما البعض. ففي كل مرة تستخدم فيها CurrentDB فإنك تكون بذلك تطلب من أكسس إنشاء كائن جديد يشير إلى قاعدة البيانات الحالية. ومن ناحية أخرى، يستخدم Jet فقط المرجع الوحيد لقاعدة البيانات الحالية DBEngine(0)(0)، لذا لا يكون ضرورياً إنشاء كائن جديدز وعند استخدام CurrentDB يكون إنشاء المرجع الجديد أقل سرعة منه عن استخدام DBEngine(0)(0)، غير أن CurrentDB تتزامن دائماً مع وجهة المستخدم. فعلى سبيل المثال، عند إنشاء جداول جديدة فإن تظهر على الفور في نافذة Database، غير أنه عند استخدام DBEngine(0)(0) فستحتاج إلى تنفيذ طريقة RefreshDatabaseWindow لكائن Application لتحديث نافذة Database، فنصل بذلك إلى نتيجة ألا وهي أن استخدام CurrentDB قد يكون أكثر سرعة من غيره.

عند استخدام وظيفة CurrentDB للإشارة إلى كائن قاعدة البيانات، فستكون هناك طرق أربع للإشارة إلى كائن في مجموعة وهي كالآتي:

نقطة تعجب	CurrentDB!QueryDefs!Invoices
فهرسة حسب الاسم	CurrentDB.QueryDefs("Invoices")
فهرسة باستخدام متغير	StrName = "Invoices"CurrentDB.QueryDefs(strName)
فهرسة حسب الرقم	CurrentDB.QueryDefs(0)

ملاحظـــة تستخدم الأمثلة المذكورة في الفصلين ٥، ٦ تطبيق النمــوذج Northwind، وقد أنشأت بالفعل في الفصل ٥ نسخة تحمــل اســم Northwind_Ch5,6 والتي ستستخدمها مجدداً في هذا الفصل.

افتح Northwind_Ch5,6 واعرض نافذة Immediate بالضغط على Ctrl+G، ثم اخــتر المراجع التالية:

- أكتب ?CurrentDB.Name واضغط Enter، وسيتم استرجاع مسار قــاعدة البيانــات المفتوحة.
- أكتــب ?CurrentDB!Employees!Title.Required ، واضغــط Enter وســـــيتم استرجاع False.
- أكتب = CurrentDB!Employees!Title.Required واضغط Enter لاختيار إعــداد الخاصية، افتح جدول Employees وحاول حفظ سجل جديد بدون إدخال قيمته في حقل Title.

فتح قاعدتي بيانات في نفس الوقت

عند العمل بطريقة تفاعلية أو عند استخدام برمجة ماكرو فقط، يمكن فتح قاعدة بيانــات واحــدة، لكن عن العمل مع برمجة VBA يمكن فتح قواعد بيانات عديدة في نفس الوقت، بل يمكن أيضــاً إنشاء مسافات عمل منفصلة، على أن يحرس كل مسافة عمل أذون الأمن الخاصة بها، كما يمكن فتح عدة قواعد بيانات مع كل مسافة عمل "ستطرق في الفصل ١١ إلى مشكلة ما يتطلــب حلــها فتح قاعدة بيانات أخرى".

يمكن استخدام طريقة OpenDatabase لكائن Workspace لفتح قاعدة بيانات محددة غــير أن فتح قاعدة بيانات أخرى لا يكون مثل فتح كتاب عمل أخر في إكسل أو وثيقة أخرى في وورد نظراً لأنه لا يمكن رؤية قاعدة البيانات الأخرى. وبالنسبة لأكسس فهو يفتـــح قــاعدة البيانــات الأخرى في الذاكرة، فلا يكون هناك أي تمثيل مرئي. لكن مع برمجة VBA يمكن العمل مع قاعدة بيانات أخرى غير مرئية حتى ولو كانت مرئية في وجهة أكسس.

يتحدد طريقة OpenDatabase قاعدة البيانات المفتوحة لتصبح كــائن Database. عنــد استخدام طريقة يتحدد شيئاً، يمكن تعيين النتيجة في متغير ونظراً لأن طريقــة OpenDatabase تعيد كائناً، قم بتعيين النتيجة إلى متغير كائن "object variable". وستكون عبارة التعيين لمتغير الكائن كالآتي:

```
Set objectvariable = object
```

وتوضح كلمة Set الأساسية تعيين الكائن في متغير كائن ويتناول الفصل ٨ كيفية معالجة الكائن تناولاً مفصلاً.

ويكون بناء جملة طريقة OpenDatabase كالآتي:

```
Set objectvariable = workspace.OpenDatabase(dbname,options,read-
only, connect)
```

حيث:

- Workspace هو مرجع اختياري لكائن Workspace الذي يحتوي على قاعدة البيانات. لاستخدام كائن Workspace الافتراضي، أحذف المرجع.
- Dbname هو تعبير سلسلة مطلوب وهو اسم لملف قاعدة بيانات موجود بالفعل ترغب في فتحه.
- Options هي وسيطة اختيارية تقوم بإعداد خيارات مختلفة لقاعدة البيانات مثل إذا كنت تفتح قاعدة البيانات لتشغيل خاص أم مشترك.
- Read-only هي وسيطة اختيارية أخرى، وتكون True إذا كان الهدف من فتح قاعدة البيانات هو القراءة فقط، وFalse إذا كان الهدف من فتح قاعدة البيانات هو القراءة والكتابة، وتكون القيمة الافتراضية أيضاً False.
- Connect هو تغيير سلسلة اختياري لتحديد معلومات الاتصال مثل كلمات المرور على سبيل المثال.

عدد استدعاء طريقة OpenDatabase. يفتح أكسس قاعدة البيانات ثم يضيفها تلقائياً إلى مجموعة Databases.

وكمثال على ذلك، سنفتح نموذجاً أخر لقاعدة البيانات أكسس Expenses.mdb وفي نافذة Immediate اتبع الخطوات التالية:

١- أكتب:

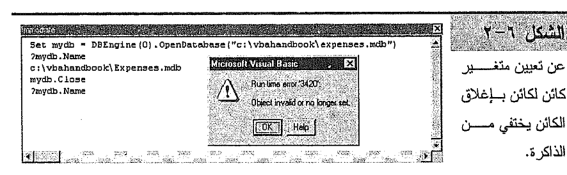
```
Set mydb=
DBEngine(0).OpenDatabase("c:\vbahandbook\expenses.mdb")
```

واضغط Enter "هذا هو مسار Expenses.mdb المستخدم في الفصل. عند حفظ قاعدة البيانات تلك في مجلد أخر، قد تحتاج إلى إدخال مسار آخر". وعلى الفور يفتح أكسس قاعدة البيانات. لكن كيف يمكن التأكد من أن قاعدة البيانات مفتوحة؟ صحيح أن متغير الكائن mydb يشير إلى قاعدة البيانات المفتوحة، لكن التأكد من أن قاعدة البيانات مفتوحة بالفعل باسترداد قيمة خاصية Name لقاعدة البيانات، أي باسترداد mydb.Name.

٢- أكتب ?mydb.Name واضغط Enter وستعرض نافذة Immediate اسم المسار بالكامل الخاص بقاعدة البيانات Expenses.mdb وبعد التأكد من أن قاعدة البيانات الثاني مفتوحة، أغلقها باستخدام طريقة Close لكائن Database عند استخدام طريقة Close، يتم إغلاق قاعدة البيانات المفتوحة ثم يتم إزالتها من مجموعة Database.

٣- أكتب mydb.Close واضغط Enter، وسيغلق أكسس قاعدة البيانات Expenses.mdb. للتأكد من أنها قد تم إغلاقها بالفعل اتبع الخطوة التالية.

٤- أكتب ?mydb.Name واضغط Enter ستخبرك الرسالة التي تنم عن وجود خطأ ما، والموضحة بالشكل ٦-٢ بأن متغير الكائن mydb غير صالح أو أنه لم يتم إعداده أصلاً وذلك نظراً لن قاعدة البيانات التي كان يشير إليها متغير mydb لم تعد موجودة بالذاكرة.



الشكل ٦-٢ عن تعيين متغير كائن لكائن بإغلاق الكائن يختفي من الذاكرة.

كائنات تشغيل بيانات جديدة

إذا أردت الإشارة إلى كائن تشغيل البيانات في إجراء VBA ولم يكن موجوداً، فستحتاج إلى إنشائه. وبوجه عام، تبنى عملية إنشاء كائن بتشغيل بيانات جديدة على ثلاث خطوات:

- قم بإنشاء الكائن مستخدماً طريقة ...Create الخاصة بالكائن الأصلي
- عرف خصائص الكائن الجديد بإعداد خواصه. وهناك العديد من خصائص الكائن التي لا يمكن إعدادها إلا عن إنشاء الكائن الجديد، ويكونون القراءة فقط وذلك بعد حفظ الكائن في قاعدة البيانات وفي بعض الحالات، يجب عليك أولاً إنشاء كائنات فرعية للكائن الجديد، فعلى سبيل المثال، عند إنشاء جدول جديد يجب إنشاء حقل واحد على الأقل قبل حفظ الجدول.
- أضف الكائن إلى المجموعة الخاصة به مستخدماً طريقة المجموعة Append.

وبوجه عام، الكائن الذي يتم إنشاؤه في الذاكرة يحفظ في قاعدة البيانات فقط عند إضافته إلى المجموعة الخاصة به لكن هناك بعض الاستثناءات:

كـائن **Workspace**: عنـد إنشـاء كـائن Workspace جديـد مسـتخدماً طريقـة CreateWorkspace لكائن DBEngine. لن تحتاج إلى إضافته قبل اسـتخدامه الكائنـات Workspaces كائنات مؤقتة لا يمكن حفظها على قرص. غير أنه عند الإشارة إلـى كـائن Workspaces الجديد خلال مجموعة Workspaces، ستحتاج إلى إضافته إلى المجموعة.

كائن **Database**: عند إنشاء كائن Database جديد مستخدماً طريقة CreateDatabase لكائن Workspace، فإن قاعدة البيانات الجديدة تضاف تلقائياً إلـى مجموعـة Databases وتحفظ على القرص.

كائن **QueryDef**: عند إنشاء كائن QueryDef جديد مستخدماً طريقة CreateQueryDef لكائن Database، فإن الاستعلام الجديد يضاف تلقائياً إلى مجموعة QueryDefs ويحفـظ على القرص.

كائن **Recordset**: عند إنشاء كائن جديد مستخدماً طريقـة OpenRecordset للكائنـات Database أو TableDef أو QueryDef أو Recordset، فإن كائن Recordset الجديـد يضاف تلقائياً إلى مجموعة Recordsets "بعد الاسم OpenRecordset قيمة خاطئة وذلـك لأنك تقوم بإنشاء كائن Recordset جديد، لا بفتح كائن Recordset موجود بالفعل".

تعد طريقة ...Create الكائن الذي تم إنشاؤه. ويتضمن بناء الجملة لكل طريقـة ...Create كلمة Set الأساسية وذلك لتعيين نتيجة الطريقة لمتغير الكائن.

إنشاء قواعد البيانات

يكون بناء الجملة لطريقة CreateDatabase كالآتي:

```
Set database = workspace.CreateDatabase(name, locale, options)
```

حيث أن:

- **Workspace**: هو مرجع لكائن Workspace الموجود "الظاهر" والتي ستحتوي على قاعدة البيانات. لاستخدام كائن Workspace الافتراضي احذف المرجع.
- **Name**: هو تعبير سلسلة، يكون له على الأكثر ٢٥٥ حرف، وهو اسم ملـف قـاعدة البيانات الذي تقوم بإنشائه، ويمكن تحديد مسار واسم ملف لا يمكن إنشاء ملف mdb إلا بهذه الطريقة.
- **Locale**: هو تعبير سلسلة مطلوب يحدد اللغة التي سوف تستخدم الـترتيب التصنيفـي لقيم النص. اسـتخدم dbLangGeneral لتحديـد الإنجليزيـة والألمانيـة والفرنسـية والبرتغالية والإيطالية والأسبانية الحديثة.

- **Options:** هو عدد صحيح اختياري يحدد تنسيق ملف محرك قاعدة بيانات Jet كما يحدد ما إذا كان سيتم تشفير قاعدة البيانات.

كمثال، ستقوم بإنشاء قاعدة بيانات جديدة فارغة، باستخدام طريقة CreateDatabase لكائن Workspace وباستخدام newdb كمتغير كائن وفي نافذة Immediate أدخل:

```
Set newdb =
DBEngine.Workspaces(0).CreateDatabase("c:\mynew.mdb" _
,dbLangGeneral)
```

سيتم إنشاء قاعدة بيانات جديدة فارغة، وستحفظ على قرص للتحقق من ذلك اختر File ⇦ Open Database ثم حدد Mynew.mdb وسيتم فتح قاعدة البيانات الفارغة الجديدة.

ملاحظة لإنشاء مشروع جديد بطريقة برمجية، لا يمكن استخدام كائن Workspace من نوع DAO، ويكون بناء الجملة لإنشاء مشروع جديد في نافذة Immediate كالآتي: Application.CreateAccessProject(projname, [connection] حيث تشير projname إلى إدخال مسار الملف كمتغير سلسلة، وتشير connection إلى سلسلة الاتصال الصحيحة لمشروع أكسس. وسيتم تناول سلاسل connection ونموذج DAO لاحقاً في هذا الفصل.

إنشاء جداول

عادة ما يكون للكائن الذي تقوم بإنشائه كائنات أخرى ثانوية. فعلى سبيل المثال، لا يمكنك إنشاء جدول دون أن يتوفر لك حقل واحد على الأقل. كذلك الحال بالنسبة لكائنات Index و Relation التي يتطلب إنشاؤها حقلاً واحداً على الأقل. وفي هذه الحالات يتطلب إنشاء الكائنات الأساسية خطوات إضافية، وكمثال على ذلك، انظر الخطوات التالية التي تتبع عند إنشاء كائن TableDef جديد يحمل اسم myShipping وذلك باستخدام حقل يحمل اسم ShipperID. يمكن إنشاء الجدول النموذج مستخدماً نافذة Immediate.

١- لإنشاء كائن TableDef جديد باستخدام طريقة CreateTableDef لكائن Database أكتب Set tdfShip = CurrentDB.CreateTableDef("myShipping") ثم اضغط Enter.

٢- لإضافة كائنات Field باستخدام طريقة CreateField لكائن TableDef، أكتب Set fldID = tdfShip.CreateField("ShipperName", dbText, 40) ثم اضغط Enter.

٣- لإضافة كائنات Field إلى مجموعة Fields أكتب tdfShip.Fields.Append fldID ثم اضغط Enter.

٤- أضف كائن TableDef الجديد إلى مجموعة TableDefs. أكتب Type CurrentDB.TableDefs.Append tdfShipثم اضغط Enter.

٥- قم بتحديث نافذة قاعدة البيانات. أكتب RefreshDatabaseWindow ثم اضغط Enter.

عند فتح نافذة Database ونقر زر Tables، ستلاحظ أن هناك جدول myShipping جديد. افتح هذا الجدول وستلاحظ أن هناك حقلاً يحمل اسم ShipperName.

إنشاء كائنات QueryDefs

ستحتاج كثيراً إلى إنشاء استعلامات جديدة في إجراء VBA، ويتم إنشاء الاستعلام الجديد على أنه كائن QueryDef. ويكون بناء جملة طريقة CreateQueryDef لكائن Database كالآتي:

```
Set qdf = database.CreateQueryDef(name,sqltext)
```

حيث أن

- **Qdf**: هو متغير كائن لكائن QueryDef الجديد.
- **Database**: هو مرجع لكائن Database المفتوح والذي سيحتوي على كائن QueryDef الجديد "في مسافة العمل ODBCDirect استخدم بدلاً مما سبق مرجعاً لكائن Connection المفتوح".
- **Name**: هو تعبير سلسلة اختياري يقوم بتسمية كائن QueryDef الجديد.
- **Sqltext**: هي عبارة SQL صالحة اختيارية، تعتبر سلسلة يعرف كائن QueryDef.

في حالة عدم تحديد وسائط name وsqltext لطريقة CreateQueryDef، يمكن استخدام عبارات التعيين لتحديد خصائص Name وSQLText لكائن QueryDef الجديد.

يتطلب إنشاء استعلامات جديدة في إجراء VBA معرفة كيفية كتابة عبارات SQL. لكن في حقيقة الأمر، إذا لم تكن على دراية كافية بعبارات SQL، فإنه يمكن تأجيل تعلمها قليلاً، واستبدلها بإنشاء نموذجاً لاستعلام جديد في عرض Design للاستعلام ثم الانتقال إلى عرض SQL ولصق عبارة SQL المطابقة في وسيطة sqltext. وكمثال على ذلك، سنقوم بإنشاء استعلام جديد يعرض

السجلات في جدول Employees وبفرزها حسب LastName، وتكون الخطوات المتبعة كالآتي:

١- في عرض Design للاستعلام، قم بإنشاء استعلام جديد يستند إلى جدول Employees. استخدم طريقة العلامة النجمية لتضمين جميع الحقول اسحب حقل LastName إلى الشبكة، ثم الغي تحديد مربع الاختيار Show لإخفاء الحقل وأدخل Ascending في خلية Sort.

٢- انتقل إلى عرض SQL، وانقل عبارة SQL التالية في Clipboard

```
SELECT Employees.*
FROM Employees
ORDER BY Employees.LastName;
```

٣- أكتب الآتي في نافذة Immediate، والصق عبارة SQL الموجودة في Clipboard وضع العبارة بين علامات تنصيص مزدوجة:

```
Set myquery = CurrentDB.CreateQueryDef("EmployeeSort", "SELECT
Employees.* FROM Employees ORDER BY Employees.LastName;")
```

أكتب العبارة بأكملها على سطر واحد. وبالضغط على Enter، يتم إنشاء الاستعلام وحفظه على قرص.

٤- اكتب RefreshDatabaseWindow ثم اضغط Enter. ستقوم نافذة Database بالتحديث لعرض الاستعلام الجديد، Employee Sort.

ملاحظة إذا كان الاستعلام الذي تريد إنشائه استعلام SQL من نوع محدد لا يمكن إنشاؤه في عرض Design للاستعلام، فهذا يعني أنك ستحتاج إلى إنشاء عبارة SQL مباشرة انظر Access 2000 Developer s Handbook by Paul Litwin, Ken Getz, and Mike Gilbert للتعرف على بعض الخطوط الإرشادية التي قد تساعد في عملية إنشاء عبارات SQL.

إنشاء مجموعة سجلات من نوع DAO

تعد مجموعة السجلات أكثر أنواع كائنات تشغيل البيانات شيوعاً التي تقوم بإنشائها في إجراءات VBA. ستلاحظ في الجدول ٦-١ أن كلاً من كائن TableDef وQueryDef لا يمثلان البيانات المخزنة في جداول قاعدة البيانات، بينما تتوفر قيم البيانات فقط كإعداد الخاصية Value لكائن Field لكائن Recordset. لذا عند معالجة البيانات في VBA باستخدام كائنات تشغيل البيانات فإنك تكون بذلك تعمل مع كائنات Recordset.

وقد كان العمل في الفصل ٤ مع مجموعة السجلات كمصادر للبيانات للنماذج. ويمكن تعريف مجموعة السجلات بأنها مجموعة من السجلات الموجودة بأحد الجداول، أو مجموعة من السجلات تنتج عن تشغيل استعلام أو عبارة SQL تنتج سجلات. وتمثل كائن تشغيل البيانات Recordset مجموعة سجلات، ولهذا الكائن أكثر من ثلاثين خاصية وعشرين طريقة للعمل مع البيانات، كما تكمن فيه كل قوة كائنات Jet لتشغيل البيانات في فرز والبحث عن وتحديث وإضافة وحذف البيانات.

أنواع مجموعات السجلات

تتوفر أربعة أنواع لكائن Recordset في نموذج DAO (بينما يتوفر نوع خامس من مسافات العمل ODBCDirect)وتختلف هذه الأنواع الأربعة بعدة طرق كما أن لكل منها عرض مختلف عن الأخر تستخدم لأجله، ولكل منها مزايا وعيوب وتوضح المقاطع التالية ميزات كل نوع من الأنواع الأربعة.

نوع الجدول Table: يمثل هذا النوع من كائن Recordset مجموعة من السجلات تمثل جدولاً واحداً في ملف mdb لقاعدة البيانات المفتوحة، لكن ليس فس جدول مرفق أو جدول ODBC. يمكن استخدام هذا النوع لاسترداد أو إضافة أو تحديث أو حذف سجلات، فباستخدامه تكون بذلك تستخدم الجدول مباشرة والميزة الأساسية التي يمتاز بها هذا النوع على غيره من الأنواع هو أنه يمكن فهرسته ومن ثم يقدم طريقة سريعة لوضع البيانات ويتوفر هذا النوع من الكائنات في مساحات عمل Microsoft Jet فقط.

نوع المجموعة الحيوية "Dynaset": "أو كائن المجموعة الحيوية" يتمثل هذا النوع في مجموعة حيوية من السجلات تمثل جدولاً في قاعدة بيانات مفتوحة، أو جدول مرفق، أو نتيجة تشغيل استعلام أو عبارة SQL SELECT إذا كان الاستعلام قابل للتحديث، فإنه يمكن استخدام هذا النوع من الكائنات لاسترداد أو إضافة أو تحديث أو حذف سجلات. عن استخدام كائن Recordset من نوع المجموعة الحيوية، فإن الكائن سيكون فقط من مجموعة من المراجع أو القيم الأساسية، ويمكن استرداد السجل الكامل فقط عند الحاجة إليه للتحرير أو العرض. صحيح أن هذا النوع يعد أقل سرعة من النوع السابق، إلا أنه أكثر مرونة في استخراج البيانات من أكثر من جدول ومن الجداول المرفقة، غير أنه لا يمكن فهرسته. وفي بعض الأحيان يكون الاستعلام الناتج غير قابل للتحديث.

الكائن Snapshot: يعد هذا النوع نسخة ثابتة من مجموعة سجلات، مستند إلى جدول أو استعلام أو عبارة SQL SELECT يمكن استخدام هذا النوع لاسترداد بيانات أو لإصدار تقارير وإذا أردت فقط المرور بالسجلات، يمكنك إنشاء كائن snapshot

للتمرير الأمامي. وفي هذا النوع يتكون الكائن من نسخة من السجل بالكامل "لكن مراجع فقط لحقول Memo وOLE Object". لا يمكن فهرسة هذا النوع كما أن البيانات غير قابلة للتحديث، غير أن هذا النوع يعد أسرع من نوع المجموعة الحيوية "dynaset".

كائن **Forward-Only**: هو يشبه كائن snapshot في كل شيء فيما عدا أنه لا يمكن التمرير في السجلات سوى للأمام لكن هذا النوع يتغير بأداء أفضل من نوع snapshot لكنه لا يمكن القيام سوى بتمريرة واحدة في مجموعة السجلات.

إنشاء كائن مجموعة السجلات "Recordset"

يمكن معالجة البيانات في إجراء VBA بإحدى الطريقتين:

- إما بفتح نموذج منضم إلى بيانات ثم استخدام كائنات Application لمعالجة البيانات.
- أو إنشاء كائن Recordset في الذاكرة يمثل البيانات ثم استخدام كائنات تشغيل البيانات لمعالجة البيانات.

مع استخدام كائنات Application ستكتب تعليمات برمجية قليلة وذلك لأن هذه الكائنات ستوفر عليك الكثير وتقوم بمعظم الأعمال "انظر الفصل ١٢ للتعرف على نماذج لإجراءات لكلتا الطريقتين". أما مع استخدام كائنات تشغيل البيانات، فإنك ستستخدم تعليمات برمجية أكثر، سيكون لديك فرصة أكبر لتحديد ومتى تريد أن ينفذ Jet كل خطوة في أي عملية لذا سيركز هذا الفصل على استخدام كائنات تشغيل البيانات.

يمكن استخدام طريقة OpenRecordset لكائن Database لإنشاء كائن Recordset جديد في جدول موجود بالفعل أو استخدام أو عبارة SQL التي تعيد السجلات بعد ذلك ستضيف طريقة OpenRecordset كائن Recordset الجديد تلقائياً إلى مجموعة Recordsets. ويكون بناء الجملة كالآتي:

```
Set rst = database.OpenRecordset (source,type,options,lockedits)
```

حيث أن:

- **Rst**: هو متغير الكائن.
- **Database**: هو مرجع لكائن Database موجود بالفعل تود استخدامه.
- **Source**: هي سلسلة تحدد اسم الجدول أو اسم الاستعلام أو تحدد عبارة SQL SELECT التي تعيد السجلات. بالنسبة لكائنات Recordset من نوع الجدول يجب أن يكون المصدر "source" اسم جدول في قاعدة البيانات.

- **Type**: هو عدد صحيح اختياري، أو ثابت حقيقي يمثل النوع كالآتي "عند تحويل جدول في قاعدة البيانات الحالية، يكون الكائن الافتراضي من نوع الجدول، اما عند تحديد جدول مرفق أو استعلام أو عبارة SQL فيكون الكائن الافتراضي من نوع المجموعة الحيوية dynaset".

DbOpenTable	كائن الجدول Table
DbOpenDynamic	الكائن الحيوي Dynamic "فقط مساحات ODBCDirect"
DbOpenDynaset	كائن المجموعة الحيوية Dynaset
DbOpenSnapshot	كائن Snapshot
DbOpenForwardOnly	كائن ForwardOnly

- **Options**: هي مجموعة ثوابت مدمجة اختيارية تحدد خصائص الكائن الجديد.
- **Lockedits**: الثوابت اختياري يحدد تأمين مجموعة السجلات.

كما يمكن إنشاء كائن Recordset جديد مستنداً إلى كائن TableDef أو كائن QueryDef. بل ويمكن أيضاً إنشاء كائن Recordset أخر موجود بالفعل ويكون لهذه الكائنات طرق OpenRecordset الخاصة بها، ويكون لها بناء الجملة التالي:

```
Set rst = object.OpenRecordset (type,options,lockedits)
```

Object هو كائن TableDef أو كائن QueryDef أو كائن Recordset أما عن باقي أجزاء الجملة فهي كما ذكر سلفاً.

ويمكن استخدام نافذة Immediate لإنشاء مجموعات السجلات التالية:

- أكتب Set rstEmployees = CurrentDB.OpenRecordset("Employees", dbOpenDynaset) ثم اضغط Enter لإنشاء كائن المجموعة الحيوية (dynaset) في جدول Employees.
- أكتب Set rstCategories = CurrentDB.OpenRecordset("Categories") ثم اضغط Enter فنشاء كائن الجدول (table) في جدول Categories.
- أكتب Set rstSuppliers = CurrentDB.OpenRecordset("SELECT * FROM Suppliers ORDER BY [CompanyName]", dbOpenSnapshot) ثم اضغط Enter لإنشاء كائن snapshot في عبارة SQL.

- أكتب Set rstCustomers = CurrentDB!Customers.OpenRecordset ثم اضغط Enter لإنشاء كائن جدول (table) في جدول Customers.
- أكتب Set rstSales = CurrentDB.QueryDefs("Sales by Category").OpenRecordset ثم اضغط Enter لإنشاء كائن المجموعة الحيوية (dynaset) في استعلام Sales by Category.

ملاحظة عند فتح مجموعة سجلات بجدول في قاعدة البيانات "في مساحة عمل مايكروسوفت Jet" مع عدم تحديد نوع يقوم Jet بإنشاء كائن من نوع الجدول "table". وعند فتح استعلام أو جدول مرتبط مع عدم تحديد النوع، يقوم Jet بإنشاء كائن من نوع specify.

إغلاق مجموعة سجلات

كل كائن Recordset تقوم بإنشائه في إجراء VBA يظهر فقط أثناء تشغيل هذا الإجراء، الذي بانتهائه تختفي كائن Recordset. وإذا أردت إغلاق كائن Recordset أثناء تشغيل الإجراء، استخدم طريقة Close، التي تقوم بإغلاق كائن Recordset. فعلى سبيل المثال، إذا أردت إغلاق rstSales أكتب rstSales.Close في نافذة Immediate ثم اضغطEnter

ملاحظة ليس من الضروري إغلاق كائنات Recordset بطريقة واحدة قبل انتهاء الإجراء، لكن إن فعلت ذلك، ستجعل تعليماتك البرمجية أكثر سهولة في فهمها.

معالجة كائن DAO من نوع Recordset

عند إنشاء كائن Recordset. فإنك تكون بذلك تضع صفوفاً من البيانات في ذاكرة مؤقتة، فلا تعرض تلك الصفوف على الشاشة وفي المرة الواحدة يمكن الإشارة إلى صف واحد، ويعرف هذا الصف المشار إليه باسم السجل الحالي "current record". هذا السجل الحالي هو الوحيد من نوعه الذي يمكن تعديل أو استبدال البيانات منه وعند الإشارة إلى حقول في كائن Recordset. ستحصل على القيم من هذا السجل الحالي أيضاً. لكن تذكر جيداً أنه في كل مرة لا يمكن أن يكون سوى سجل واحد فقط. عند إنشاء كائن Recordset لأول مرة مستخدماً طريقة OpenRecordset. فإن أول سجل يكون هو السجل الحالي، وذلك بالطبع في حالة وجود

سجلات حيث أنه في بعض الأحيان لا يكون لكائن Recordset سجل حالي وذلك في حالة عدم وجود سجلات لمجموعة السجلات.

ونظراً لأنه لن يتوافر سوى سجل واحد فقط في المرة الواحدة، فإنك ستحتاج دائماً إلى طرق للتنقل من سجل إلى أخر متخذاً السجل الذي تنتقل إليه. سجلاً حالياً بحيث يمكنك العمل معه. وهناك نوعان أساسيان للتنقل هما التنقل الحقيقي "physical navigation" والتنقل المنطقي "logical navigation" في أكسس، يستخدم التنقل الحقيقي وذلك عند نقر أزرار التنقل في الركن السفلي الأيسر لنموذج أو ورقة بيانات لنقلها من سجل إلى أخر تبعاً لموضعها الحقيقي في مجموعة السجلات لكن عند استخدام مربع حوار Find "اختر Edit ⇦ Find" وإدخال معيار البحث في مربع النص Find What، فإنك تكون بذلك تستخدم التنقل المنطقي للانتقال مباشرة إلى السجل الأول الذي يطابق المعيار ولكائن Recordset طرق وخصائص لكل النوعين من التنقل.

استخدام التنقل الحقيقي

يمكن الانتقال من سجل إلى أخر تبعاً للموضع الحقيقي بإحدى الطريقتين: إما باستخدام طرق ...Move لتكرار تأثير أزرار التنقل في أكسس، أو بحفظ موضعك في مجموعة السجلات عن طريق إعداد إشارة مرجعية ثم الرجوع فيما بعد إلى نفس السجل.

استخدام طرق ...Move

يمكن الانتقال من سجل إلى أخر تبعاً للموضع الحقيقي للسجل في مجموعة السجلات باستخدام طرق ...Move لكائن Recordset. فبالنسبة لطرق MoveFirst, MoveLast, MoveNext, and MovePrevious فهي تنقل موضع السجل الحالي إلى السجل الأول أو الأخير أو التالي أو السابق في كائن Recordset محدد ويكون بناء الجملة لطرق ...Move كالآتي:

```
rst.{MoveFirst | MoveLast | MoveNext | MovePrevious}
```

Rst تشير إلى كائن Recordset object مفتوح، في كائن forward-only لا يمكنك سوى استخدام طريقة MoveNext حيث أنه لا يمكنك نقل مؤشر السجل الحالي سوى إلى الأمام تجاه السجل الأخير لمجموعة السجلات.

فعلى سبيل المثال، عند إنشاء rstEmployees في المقطع الأخير، كان السجل الأول هو السجل الحالي. وعلى أية حال، يمكنك استخدام نافذة Immediate لاختيار طرق ...Move كالآتي:

١ - أكتب ?rstEmployees.EmployeeID ثم اضغط Enter وسيظهر لك الرقم ١.

٢- أكتب rstEmployees.MoveNext واضغط Enter، ثم أكتب ?rstEmployees.EmployeeID واضغط Enter سيظهر الرقم ٢ انتقل بعد ذلك إلى السجل الأخير لترى ماذا سيحدث لو أنك حاولت استخدام طريقة MoveNext.

٣- أكتب rstEmployees.MoveLast ثم اضغط Enter، وسيصبح السجل الأخير في مجموعة السجلات هو السجل الحالي.

٤- أكتب rstEmployees.MoveNext ثم اضغط Enter، وستجد أنك قد انتقلت إلى ما بعد السجل الأخير في مجموعة السجلات. أكتب recordset. Type ?rstEmployees.EmployeeID ثم اضغط Enter، وستوضح رسالة الخطأ التي ستظهر أنه لا يوجد سجل حالي.

استخدام خصائص BOF وEOF

تستخدم خصائص BOF "بداية الملف End of File" لكائن Recordset لتحديد ما إذا كنت قد انتقلت إلى ما هو أبعد من حدود مجموعة السجلات ويكون لكلتا الخاصيتين القيمة False طالما أنك تشير إلى سجل في مجموعة السجلات. أي طالما أن هناك سجل حالي. إما في حالة الانتقال إلى ما بعد السجل الأخير، فإنه لا يكون هناك سجل حال وتكون قيمة خاصية True EOF، كذلك الحال بالنسبة لخاصية BOF عند الانتقال إلى ما قبل السجل الأول. إذا لم يكن لمجموعة السجلات أية سجلات على الإطلاق، فتكون قيمة كلتا الخاصيتين True. للتأكد من ذلك، أكتب ?rstEmployees.EOF في نافذة Immediate ثم اضغط Enter وستعاد القيمة True.

والإجراء القياسي للعمل مع مجموعة من السجلات هو إنشاء كائن Recordset واستخدام طريقة MoveNext لتنفيذ حلقة بالسجلات واحداً يلو الآخر. لتحديد متى تكون قد انتهيت. اختبر قيمة خاصية EOF في بداية كل تمرير من الحلقة التي تنفذها. كلما تجد أن القيمة ما زالت False، قم بتمرير آخر، إلى أن تصبح قيمة True EOF، فهنا تعرف أنك قد انتقلت إلى ما بعد السجل الأخير وأن الحلقة قد انتهت بالفعل. يوضح الفصل ٩ كيفية إنشاء إجراءات لتنفيذ حلقات في مجموعة السجلات.

إنشاء إشارات مرجعية

لقد علمت في الفصل ٥ أن أكسس يتعقب السجلات في مجموعة سجلات النموذج عن طريق إنشاء إشارة مرجعية لتكون سلسلة مزدوجة واحدة لكل سجل وذلك بمجرد فتح النموذج. لكن ما يقوله هذا الفصل هو أن Jet أيضاً يقوم بإنشاء إشارات مرجعية، فعند إنشاء كائن Recordset. يقوم Jet تلقائياً بتعيين مجموعة فريدة من الإشارات المرجعية في حالة الإشارة إلى سجل تريد العودة إليه فيما بعد، يمكن حفظ مكانك عن طريق حفظ الإشارة المرجعية للسجل على أنها

متغير، وللعودة إلى السجل مرة أخرى فيما بعد، قم بإعداد خاصية Bookmark للقيمة المحفوظة ويمكنك اختيار التقنية في نافذة Immediate كالآتي:

١- أكتب rstEmployees.MoveFirst واضغط Enter، ثم أكتب rstEmployees.MoveNext واضغط Enter. سيكون السجل الحالي هو السجل الثاني "EmployeeID = 2". احفظ مكانك في الخطوة التالية:

٢- أكتب strMark = rstEmployees.Bookmark ثم اضغط Enter. سيقوم المتغير strMark بتخزين الإشارة المرجعية.

٣- استخدم طريقة Move للانتقال إلى سجل آخر، ثم عد مرة أخرى إلى المكان المحفوظ.

٤- أكتب rstEmployees.Bookmark = strMark ثم اضغط Enter وللتأكد ادخل rstEmployees.EmployeeID? ثم اضغط Enter. وبذلك تكون قد عدت إلى السجل ذي الإشارة المرجعية.

استخدام طريقة Move

يمكن استخدام طريقة Move لنقل موضع السجل الحالي إلى الأمام "تجاه السجل الأخير" أو إلى الخلف "تجاه أول صف" لعدد محدد من الصفوف. بل ويمكن تحديد سجل معين تود الانتقال منه. ويكون بناء الجملة لطريقة Move كالآتي:

```
rst.Move row,start
```

حيث:

- Rst هو مرجع كائن Recordset.
- Rows هو عدد صحيح طويل يوضح عدد الصفوف، فإذا كان العدد إيجابي، انتقل إلى الأمام، فيما عدا ذلك انتقل إلى الخلف وبالنسبة لكائن forward-only يجب أن يكون rows عدداً إيجابياً.
- Start هو متغير سلسلة اختياري يعرف الإشارة المرجعية.

يمكن اختبار طريقة Move في نافذة Immediate أكتب rstEmployees.Move 4 ثم اضغط Enter. تأكد من أنك الآن بالسجل باستخدام EmployeeID = 6.

استخدام التنقل المنطقي

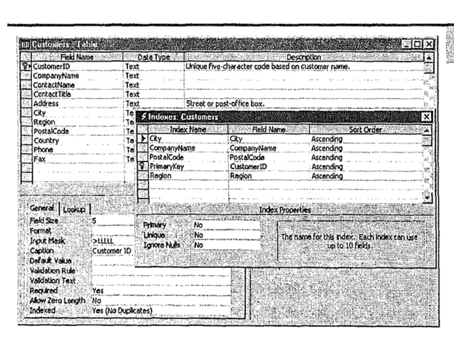
إذا أردت إدراج سجل يرضي شرط البحث، ستعتمد التقنية التي تستخدمها على نوع كائن Recordset الذي قمت بإنشائه فإذا كنت تعمل مع كائن table. يمكنك الاستفادة من الفهارس واستخدام طريقة Seek. وإذا كنت تعمل مع كائن dynaset أو snapshot، فلن تكون الفهارس

ملائمة وستستخدم طرق Find. وإذا ما نجحت أي من تقنيتي البحث هاتين في إيجاد سجل مطابق لشرط البحث، يكون السجل الذي تم إيجاده هو السجل الحالي، إذا لم يتم تحديد موقع أي سجل فهذا يعني أنه لا يوجد أي سجل حالِ تعد تشغيل الطريقة لتحديد ما إذا كان البحث قد نجح في مهمته، اختر خاصية NoMatch لكائن Recordset. إذا كان البحث قد نجح فسيتم العثور على مطابق وتكون قيمته NoMatch False، أما إذا أخفق البحث في مهمته، فلن يكون هناك مطابق وستكون قيمة NoMatch True.

استخدام طريقة Seek

تستخدم طريقة Seek الفهارس لتحديد موضع السجلات بأسرع طريقة ممكنة وتستخدم هذه الطريقة فقط مع كائن table حيث يستند شرط البحث إلى قيم بالفهرس لكن قبل استخدام طريقة See، يجب أن يكون للجدول الذي تعمل معه فهرس واحد على الأقل، ويجب أن تقوم بإعداد خاصية Index لكائن Recordset على حسب الفهرس الذي تريد استخدامه في عملية البحث. فعلى سبيل المثال، يكون لجدول Customers الفهارس الموضحة في الشكل ٦-٣.

يمكنك استخدام واحداً من هذه الفهارس أو إنشاء كائن Index جديد انظر الفصل ١٤. تعرف قيمة خاصية Index باسم الفهرس الحالي "current index".



الشكل ٦-٣

الفهارس المستخدمة لجدول Customers.

ويكون بناء جملة طريقة Seek كالآتي:

```
tablerecordset.Seek comparison, key1, key2... key13
```

حيث:

- Tablerecordset: هو مرجع لكائن table موجود بالفعل يحدد فهرسة الحالي خاصية Index لكائن Recordset.
- Comparison هو تعبير سلسلة يشتمل على أحد عوامل المقارنة التاليـة، < أو <= أو = أو => أو > ويجب اتباع كل عامل تشغيل بفاصلة.
- Key1, key2... key13 هي قيم لحقول في الفهرس الحالي من رقم ١ إلى رقم ١٣.

فعلى سبيل المثال، للبحث عن عملاء لمدينة معينة، اتبع الخطوات التالية:

١- لإعداد الفهرس الحالي City، أكتب "rstCustomers.Index = "City ثـم اضغط Enter.

٢- لتحديد موضع أول تحميل من لندن، أكتب "rstCustomers.Seek "=", "London ثم اضغط Enter.

٣- لتحديد ما إذا كان البحث قد نجح في مهمته أكتـب ?rstCustomers.NoMatch ثـم اضغط Enter. إذا كان البحث ناجحاً، سيتم إعادة القيمة False.

٤- أكتب ?rstCustomers.CompanyName ثم اضغط Enter. سيتم إعـادة اسـم أول يحمل ويكون كالآتي .Around the Horn.

استخدام طرق ...Find

تعمل طريقة Seek مع كائن table فقط، لذا عندمـا يكـون الكـائن مـن نـوع dynaset أو snapshot استخدام طرق ...Find، تلك الطرق التي لا يمكن تطبيقها على كائنات table.

يمكن تكرار تأثير أمر Find في أكسس باستخدام طرق ...Find لكائن Recordset. بالنسبة لطرق FindFirst أو FindLast أو findNext أو FindPrevious فهي تحدد موضـع السـجل الأول والأخير والتالي والسابق، الذي يرضي المعيار المحدد ويجعل من هذا السـجل المعـالج، سجلاً حالياً ويكون بناء جملة طرق ...Find كالآتي:

```
rst.{FindFirst | FindLast | FindNext | FindPrevious} criteria
```

حيث rst هو مرجع لكائن dynaset أو snapshot موجود بالفعل، criteria هـي تعبير سلسلة لتحديد موضع السجل (التعبير هو WHERE لعبارة SQL لكن بدون كلمة WHERE).

فعلى سبيل المثال، يمكن إيجاد أول مورد من برلين كالآتي:

١- أكتب "'rstSuppliers.FindFirst "City = 'Berlin في نافذة Immediate ثم اضغط Enter.

٢- لتحديد اسم المورد أكتب ?rstSuppliers.CompanyName ثم اضغط Enter.

إضافة وتحرير وحذف سجلات

ما قد استنتجه المبرمجون ذو الخبرة في كائنات تشغيل البيانات هي قدراتها على التحكم في كل عملية، لكن ما قد وجده المبرمجون من ذوي الخبرة شيئاً ملحاً هو ضرورة التحكم في كل عملية لفهم الطرق التي يجب تضمينها في الإجراءات التي تعدل البيانات باستخدام كائنات تشغيل البيانات، يجب أن تعلم جيداً أن Jet لا يقوم تلقائياً بتنفيذ الخطوات التي تنفذ تلقائياً عند كذلك إجراءات لتعديل بيانات بالنموذج باستخدام كائن Application.

تحرير وتحديث السجلات

إذا أردت تعديل البيانات في سجل ما، يجب أولاً لحالة هذا السجل ليصبح هو السجل الحالي. ولنقل مؤشر السجل الحالي إلى السجل الذي تريد تغييره، يمكنك استخدام طرق ...Move أو طرق ...Find أو Seek. غير أن المشكلة التي سوف تواجهها عند تحرير البيانات هي أن التغييرات لا يتم أحداثها مباشرة بالسجل، بل يتم نسخة ووضعه بمكان ما بالذاكرة يعرف باسم النسخة المؤقتة "copy buffer" التي يقوم Jet بإنشائها خصيصاً لتحرير البيانات ويجب أن تتعامل تعليمات VBA البرمجية مع المحتوى الذي ينتقل إلى داخل النسخة المؤقتة أو إلى خارجها. لإعادة نسخ محتويات النسخة المؤقتة في كائن Recordset. يجب استخدام طريقة Update فإذا قمت بأداء أي عملية تنقل إلى سجل أخر أو تنهي إجراء VBA أو تغلق كائن Recordset بدون استخدام طريقة Update أولاً، فلن يتم حفظ التغييرات وسيتم تجاهل محتويات النسخة المؤقتة بدون تحذير سابق.

يمكن العمل في نافذة Immediate لتغيير البيانات في كائن rstCustomers. فستقوم بتغيير اسم الشركة من Around the Horn لتكون Around the Cape في السجل الذي تكون قد وجدته قبل ذلك مستخدماً طريقة Seek. اتبع الخطوات التالية:

١- لعمل نسخة من السجل الحالي في النسخة المؤقتة للتحرير مستخدماً طريقة Edit، أكتب rstCustomers.Edit ثم اضغط Enter.

٢- لتعيين القيم الجديدة للحقول التي تريد تغييرها. أكتب rstCustomers.CompanyName = "Around the Cape" ثم اضغط Enter، وستصبح القيمة الجديدة في النسخة المؤقتة.

٣- لحفظ التغييرات في السجل الحالي مستخدماً طريقة Update، أكتب rstCustomers.Update ثم اضغط Enter.

٤- للتأكد من التغييرات قد تم حفظها بالفعل، أكتب ?rstCustomers.CompanyName ثم اضغط Enter، وستعرض نافذة Immediate القيمة الجديدة.

ويعطي نظام تحرير البيانات مثالاً على الاختلافات الأساسية بين تعديل البيانات باستخدام وجهة أكسس، وبين تعديلها باستخدام كائنات تشغيل البيانات فعندما تعمل بطريقة تفاعلية فإنك تضع السجل الذي تريد تغييره في نموذج، ثم تكتب التغييرات في أداة تحكم، وفي اللحظة التي تنتهي فيها من كتابة الحرف الأول، يوضح قلم التحرير في محدد السجل انك تقوم بإدخال قيم جديدة في النسخة المؤقتة "يقوم التطبيق Application باستدعاء طريقة Edit من أجلك". عند الانتهاء من أحداث التغييرات اللازمة يمكن حفظها بعدة طرق، بالنقر مثلاً في سجل آخر، أو بالضغط على Shift+Enter أو بإغلاق النموذج يقوم التطبيق "Application" باستدعاء طريقة Update من أجلك.

إذا قررت عدم حفظ التغييرات بالسجل، استخدم طريقة CancelUpdate لتدفق النسخة المؤقتة.

ملاحظة لا يمكن تطبيق طريقة Edit على كائنات snapshot أو forward-only نظراً لأنها جميعاً نسخ ثابتة من السجلات.

إضافة وتحديث السجلات

كما هو الحال بالنسبة لتغييرات السجل، لا تضاف السجلات الجديدة مباشرة إلى مجموعة السجلات، بل تضاف إلى النسخة المؤقتة في الذاكرة. وتستخدم تعليمات VBA البرمجية طريقة AddNew لإضافة محتويات النسخة المؤقتة إلى مجموعة السجلات.

ويعتمد المكان الذي يتم فيه إدراج السجل الجديد على نوع مجموعة السجلات، ففي مجموعة سجلات من نوع dynaset يضاف السجل الجديد في نهاية المجموعة، وفي مجموعة سجلات من نوع table يوضع السجل الجديد في ترتيبه التصنيفي المناسب.وذلك في حالة أن خاصية Index قد تم إعدادها بالفعل، فيما عدا ذلك، يكون في نهاية مجموعة السجلات. عند إضافة سجل جديد. يقوم Jet بإنشاء إشارة مرجعية للسجل قم يقوم بتخزينها في خاصية LastModified. كل ذلك سيحدث بدون أو في تأثير على السجل الحالي الذي يظل أيضاً هو السجل الذي كان حالياً قبل إضافة السجل. إذا أردت أن يكون السجل الجديد هو السجل الحالي، انقل مؤشر السجل الحالي إليه.

كذلك يمكن العمل في نافذة Immediate لإضافة سجل جديد إلى كائن rstCustomers. اتبع الخطوات التالية:

١- لإنشاء سجل جديد في المخزن المؤقت مستخدماً طريقة AddNew لكائن Recordset. أكتب rstCustomers.AddNew ثم اضغط Enter.

٢- لتعيين قيم جديدة للحقول في المخزن المؤقت، أكتب عبارات التعيين الآتية في نافذة Immediate ثم اضغط Enter بعد كل من:

```
rstCustomers.CustomerID = "ROUND"
rstCustomers.CompanyName = "Round the Bend"
rstCustomers.City = "London"
```

٣- لإضافة المحتويات إلى مجموعة السجلات، أكتب rstCustomers.Update ثم اضغط Enter. ونظراً لأن rstCustomers هي مجموعة سجلات من نوع table ونظراً لأن هناك فهرس حال "قم قبل ذلك بإعداد خاصية Index لأن تكون London"، يتم وضع السجل الجديد مع باقي عملاء London، وسيكون السجل الحالي هو نفسه السجل الذي يحمل اسم الشركة Around the Cape.

٤- لمزيد من التأكيد أكتب ?rstCustomers.CompanyName ثم اضغط Enter.

٥- للانتقال إلى السجل الجديد، قم بإعداد خاصية Bookmark للإشارة المرجعية الجديدة للسجل المخزنة كإعداد خاصية LastModified أكتب rstCustomers.Bookmark = rstCustomers.LastModified ثم اضغط Enter.

٦- لمزيد من التأكيد أكتب ?rstCustomers.CompanyName ثم اضغط Enter.

إذا قررت عدم إضافة السجل الجديد، يمكنك استخدام طريقة CancelUpdate ليتدفق المخزن المؤقت.

حذف السجلات

عند حذف السجل، يتم وضعه بمكان ما في الذاكرة، سنطلق عليه اسم المخزن المؤقت للحذف، وهو الذي يقوم Jet بإنشائه ليشتمل على كل ما تم حذفه وجدير بالذكر يظل السجل هو نفسه السجل الحالي على الرغم من أنه لا يمكن تحريره أو استخدامه، وعند الإشارة إلى السجل المحذوف، يظهر خطأ تشغيل الوقت. لا حظ انه لا توجد طريقة CancelDelete للتراجع عن عملية ما، بل يحذف السجل ببساطة وبدون تحذير سابق.

تلميح إذا أردت عرض رسالة تأكيد حذف، يمكنك إنشاء مجموعة من العبارات تعرف باسم الإجراءات "transaction" للتراجع عن الحذف، وذلك إذا كان المستخدم قد اتخذ قراراً بعدم حذف السجل. انظر الفصل ١٣ لمزيد من المعلومات عن إنشاء الإجراءات "أو التعاملات".

كذلك يمكن العمل مع نافذة Immediate لحذف السجل الذي قمت بإضافته إلى كائن rstCustomers اتبع الخطوات التالية:

١- لوضع السجل في المخزن المؤقت للحذف مستخدماً طريقة Delete، اكتب rstCustomers.Delete ثم اضغط Enter. لمزيد من التأكيد أكتب ?rstCustomers.CompanyName وستظهر رسالة خطأ تقول أن السجل غير موجود.

٢- انتقل إلى سجل أخر ليكون هو نفسه السجل الحالي. أكتب rstCustomers.MoveNext ثم اضغط Enter. لمزيد من التأكيد أكتب ?rstCustomers.CompanyName ثم اضغط Enter.

استخدام النسخ

في فصول قادمة، ستعرف أنه في بعض الأحيان يكون هناك أكثر من مؤشر سجل حال لكائن Recordset. استخدم طريقة Close لإنشاء كائن Recordset جديد يقوم بتكرار كائن Recordset موجود بالفعل، وسيكون بناء الجملة كالآتي:

```
Set rstclone = original.Clone
```

حيث rstclone هو متغير الكائن الجديد لكائن Recordset و reference هو مرجع لكائن Recordset.

عند إنشاء نسخة، يكون لكل من الكائن المنسوخ وكائن Recordset الأصلي السجلات الحالية الخاصة به، ويمكنك التنقل في كائن Recordset المكرر نفس إرشادات كائن Recordset الأصلي المرجعية، ومن ثم يمكنك إعداد إشارة مرجعية مع السمة، واستعادة الإشارة المرجعية مع الكائن الأصلي والعكس صحيح. غير أن الكائن الأصلي والمنسوخ لا يكونان متطابقان تماماً، فعلى سبيل المثال، إذا كانت مجموعة السجلات من نوع table ولها فهرس حالي، لا يكون للنسخة نفس قيمة خاصية Index، وإذا كانت مجموعة السجلات من نوع dynaset أو snapshot، لا يكون للنسخة نفس قيم خصائص Filter وSort.

ونظراً لأن كلاً من مجموعتي السجلات الأصلية والمنسوخة يشيران إلى نفس الجدول المخزن أو نفس الاستعلام، فإنه يمكنك تعديل البيانات، أو إضافة أو حذف سجلات إما باستخدام الكائن الأصلي أو المنسوخ، وفي كلتا الحالتين، يمكنك استخدام طريقة Close بدون إغلاق الكائن الآخر.

وقد قام الفصل ٥ بالفعل بتعريف ما المقصود بخاصية RecordsetClone لكائن Form. فعند فتح نموذج منضم، يقوم Jet بإنشاء كائن Recordset مستنداً إلى الجدول أو الاستعلام أو

إلى عبارة SQL المحددة في خاصية RecordSource للنموذج. عند استخدام خاصية RecordsetClone للنموذج، فإنك تكون بذلك يقوم بإنشاء مؤشر سجل حال جديد لكائن Recordset للنموذج لا بإنشاء كائن Recordset جديد. وأهم ما تستفيده من جراء استخدام خاصية RecordsetClone في تعليمات VBA البرمجية. هو أنك لن تضطر إلى المرور بالتسلسل الهرمي للكائن Jet للوصول إلى كائن Recordset. فمع خاصية RecordsetClone يمكنك إنشاء مرجع لكائن تشغيل بيانات بدون المرور بشريط DAO الأحمر، أي أنه يمكنك إنهاء التشغيل حول تسلسل DAO الهرمي وبعد إنشاء خاصية RecordsetClone يمكنك استخدام طرق كائن Recordset مع الكائن المنسوخ.

ملاحظة لا تجعل المصطلحات المتشابهة تسبب لك أي نوع من المشكلات، فخاصية RecordsetClone لا تقوم إلا بإنشائه مؤشر سجل مستقل لمجموعة سجلات النموذج، ولكنها لا تقوم بإنشاء كائن Recordset منفصل. وعلى الجانب الأخر، تقوم طريقة Close بإنشاء كائن Recordset منفصل الذي يعد تكراراً لكائن Recordset الأصلي وله نفس إشاراته المرجعية.

نموذج DAO

يعد الجزء التالي من هذا الفصل مرجعاً لكائنات تشغيل البيانات وبالنسبة لكائن DBEngine ولكل كائن جاء في مجموعة، هناك وصف للميزات الهامة يتبعه جداول لخصائص وطرق الكائن الموثقة، لكن خاصية لكائنات تشغيل البيانات أحداث معرفة. لذا، قبل استخدام خاصية ما أو طريقة ما للمرة الأولى، يجب للإشارة أولاً إلى الشرح الكامل الموجود بالتعليمات الفورية في أكسس.

ملاحظة هناك ميزة هامة في أكسس ٢٠٠٠ وفي نموذج مايكروسوفت لكائن DAO، وهي أنه يمكنك استخدام محرك Jet لقاعدة البيانات لتشغيل البيانات أو استخدامك تكنولوجيا ODBCDirect للعمل مع ملقم من نوع ODBC مثل Microsoft SQL Server بدون تحميل Jet ويعرض هذا الجزء من الفصل العديد من الخصائص الجديدة التي تقترن بـ ODBCDirect. انظر Paul Litwin, Ken Getz, and Mike Gilbert (Sybex, 1999) لمزيد من المعلومات عن استخدام ODBCDirect.

كائن DBEngine

يستخدم كائن DBEngine لصيانة ملفات قاعدة البيانات، ولفحص تفاصيل الخطأ وأخيراً إنشاء جلسات عمل جديدة تبدأ جلسة العمل بتسجيل الدخول وتنتهي بتسجيل الخروج. في حالة عدم وجود إجراءات أمنية، يسجل أكسس الدخول والخروج تلقائياً عن تشغيل أكسس وإنهائه. لذلك فكر بجلسة عمل لتكون مسافة عمل تحرسها إجراءات أمنية قمت بإعدادها. يوضح جدول ٦-٢ وجدول ٦-٣ خصائص وطرق كائن DBEngine.

ملاحظة ما يعد مرة هامة في أكسس ٢٠٠٠ هو انه يمكنك تشغيل أوامر CompactDatabase وrepairDatabase في قاعدة بيانات مفتوحة "اختر Tools ⇦ Database Utilities".

الجدول ٦-٢: خصائص كائن DBEngine

الخاصية	التشغيل	نوع البيانات	الشرح
DefaultUser	Read/write	سلسلة ٢ من ١ إلى ٢٠ حرف	تقوم بإنشاء اسم المستخدم الذي يستخدمه Jet عند تشغيل أكسس، ويكون admin افتراضياً
DefaultPassword	Read/write	سلسلة ١٤ حرف على الأكثر: أحرف كبيرة	تقوم بإعداد كلمة المرور التي يستخدمها Jet عند تشغيل أكسس، وتكون السلسلة التي لها طول يقدر بصفر " " افتراضية
DefaultType	Read/write	عدد صحيح طويل	يوضح ما إذا كانت مسافة العمل التي تم إنشاؤها من نوع Jet أم من نوع ODBCDirect

الجدول ٦-٢: خصائص كائن DBEngine

الخاصية	التشغيل	نوع البيانات	الشرح
LoginTimeout	Read/write	سلسلة	يقوم بإعداد أو بالتوصل إلى المسار الذي يستخدم للوصول إلى مفتاح التطبيق في Regstry
IniPath	Read/write	عدد صحيح	يقوم بإعداد أو إعادة عدد الثواني قبل ظهور أي خطأ عند محاولة تسجيل الدخول في قاعدة بيانات ODBC، مثل SQL Server
Version	Read-only	سلسلة	تقوم بإعادة قيمة DAC التي تستخدم في الوقت الحالي
SystemDB	Read/write	سلسلة	تقوم بإعداد أو إعادة المسار للموضع الحالي لملف المعلومات عن مجموعة العمل

الجدول ٦-٣: طرق كائن DBEngine

الطريقة	الشرح
CompactDatabase	تقوم بنسخ وضغط قاعدة بيانات مغلقة، كما يمكنك تغيير إصدار Jet أو إعادة الترتيب أو
CreateWorkspace	تقوم بإنشاء كائن Workspace جديد
Idle	تعلق معالجة البيانات بحيث يستطيع Jet مواصلة إضافة المهام مثل الذاكرة

الجدول ٦-٣: طرق كائن DBEngine

الطريقة	الشرح
RegisterDatabase	تقوم بإدخال معلومات اتصال لمصدر البيانات ODBC.INI. يمكنك أيضاً استخدام إعداد Windows ODBC Control Panel لتوفير معلومات الاتصال اللازمة
SetOption	في وقت التشغيل، تتجاهل القيم المعدة في Registry
OpenConnection	تفتح كائن Connection في مصدر بيانات ODBC

كائنات المجموعة

لكل كائن من كائنات المجموعة لتشغيل الخمسة عشر خاصية Count التي تعيد رقم الكائنات في المجموعة، والثلاث طرق التالية:

- Append (A) تضيف كائن تشغيل بيانات جديد إلى المجموعة
- Delete (D) تحذف كائن تشغيل بيانات من المجموعة
- Refresh (R) تقوم بتحديث الكائنات في المجموعة لتعكس المخزون الصحيح لعناصر المجموعة

يضع جدول ٦-٤ قائمة بالمجموعات وبطرقها مستخدماً الاختصارات A وD وR

الجدول ٦-٤: طرق كائنات المجموعة "Collection"

المجموعة	الطرق	التعليق
Workspaces	A, D, R	لن تحتاج إلى إضافة كائن Workspace قبل استخدامه
Databases	R	يضاف كائن Database جديد تلقائياً إلى مجموعة Database لإزالة كائن Database من الذاكرة، استخدام طريقة Close الخاصة بهذا الكائن

الجدول ٦-٤: طرق كائنات المجموعة "Collection"

المجموعة	الطرق	التعليق
TableDefs	A, D, R	
Fields	A, D, R	
Indexes	A, D, R	
Relations	A, D, R	يضاف كائن Recordset تلقائياً إلى مجموعة Recordsets كما يتم إزالته تلقائياً عن إغلاق كائن Recordset كما يمكنك استخدام طريقة Close لإزالة كائن Recordset من الذاكرة
QueryDef	A, D, R	يضاف كائن QueryDefs جديد له اسم صالح تلقائياً إلى مجموعة QueryDefs. عند إنشاء كائن QueryDefs بدون اسم ثم بعد ذلك إعداد خاصية Name، يجب إضافة الكائن إلى مجموعة QueryDefs
Parameters	R	يتم إنشاء المعامل فقط في تعريف كائن QueryDefs
Errors	R	عند حدوث خطأ ما، يضيف Jet كائنات Errors تلقائياً، وعند حدوث خطأ تالي يقوم بمحو مجموعة Errors
Containers	R	تتضمن مجموعة Containers كل الكائنات Containers المعرفة من قبل أكسس وjet
Documents	R	تتضمن مجموعة Documents المعرفة من قبل أكسس وjet
Users	A, D, R	
Groups	A, D, R	
Properties	A, D, R	تستخدم هذه الطرق لإضافة أو حذف خصائص معرفة من قبل المستخدم، لكن لا يمكن حذف الخصائص المضمنة.



بطبيعة الحال، عند إضافة أو حذف كائنات من المجموعة يحتفظ محرك Jet بمخزون دقيق، غير أنه قد يخرج منه بالمخزون الحقيقي عند العمل في جو متعدد المستخدمين قد يحدث ذلك عند استخدام عبارات SQL التي تضيف الكائنات وتحذفها أو عند إضافة أو حذف كائنات في وجهة أكسس للمستخدم في مثل هذه الحالات، تأكد من أنك تعرض أحدث إصدار، وذلك مستخدماً طريقة Refresh لتحديث المجموعة. لا تقم بتحديث المجموعة إلا عند الضرورة، نظراً لأن طريقة method تعد عملية مستهلكيه للوقت.

كائن Workspace

يقوم كائن Workspace بتعريف جلسة العمل. وبالنسبة للعمليات التي تحدث أثناء جلسة العمل، فجميعها يخضع للتصريح الذي يحكمه اسم المستخدم الخاص بك وكلمة المرور. أثناء جلسة العلم يمكن إنشاء قواعد بيانات جديدة أو فتح قواعد بيانات عديدة في وقت واحد، أو إدارة الإجراءات أو إنشاء مستخدمين ومجموعات أمنية جديدة. والإجراء "أو التعامل" هو سلسلة من التغييرات التي تم إحداثها باعتبارها وحدة، وفيه يتعامل محرك قاعدة البيانات مع العمليات على أنها خبر مكتمل أولاً يتعامل معه، كما يؤثر التعامل على الجميع قواعد البيانات في جلسة العمل، فهو لا يقتصر على قاعدة بيانات واحدة بعينها.

عن تشغيل أكسس، يتم تشغيل Jet، الذي يقوم تلقائياً بإنشاء Workspace جديد إذا لم يكن لديك أمناً متاحاً فإن Jet سيقوم بفتح كائن Workspace الافتراضي الذي يعرف باسم #Workspace باعتباره اسم المستخدم الافتراضي، وبفتح السلسلة ذات الطول صفر باعتبارها كلمة المرور الافتراضية. لمزيد من التأكيد أكتب DBEngine.Workspaces(0)? في نافذة Immediate مستخدماً مجموعة DBEngine(0).Name? الافتراضية ثم اضغط Enter.

يضع الجدولان ٦-٥ و٦-٦ قائمة بخصائص وطرق كائن Workspace.

الجدول ٦-٥: خصائص كائن Workspace

الخاصية	التشغيل	نوع البيانات	الشرح
DefaultCursorDriver	Read/write	عدد صحيح طويل	يحدد نوع Cursor Driver الذي تم إنشاؤه في مساحة العمل ODBCDirect
IsolateODBCTrans	Read/write	منطقي	يشارك في أو يفضل ODBC الخاصة بكائنات Workspace العديدة التي تم فتحها من أجل بعض التعاملات التي تجرى في نفس الوقت
LoginTimeout	Read/write	عدد صحيح	يقوم بإعداد أو بإعادة عدد الثواني المراد انتظارها قبل إصدار أي خطأ عند محاولة الدخول إلى قاعدة بيانات من نوع ODBC "فقط من أجل مسافات العمل من نوع ODBCDirect"
Name		سلسلة، تتكون من عشرين حرفاً على الأكثر	تقوم بإعداد أو بإعادة الاسم المعرف من قبل المستخدم لكائن Workspace
Type	Read/Write لمسافات العمل Read,ODBC فقط لمسافات عمل Jet	عدد صحيح	تقوم بإعداد أو بإعادة قيمة ما لتوضح ما إذا كانت مسافة العمل متصلة بـ Jet أو بمصدر بيانات من نوع ODBC

الجدول ٦-٥: خصائص كائن Workspace

الخاصية	التشغيل	نوع البيانات	الشرح
UserName	Read فقط	سلسلة	تقوم بإعداد أو بإعادة قيمة ما تمثل مالك كائن Workspace

الجدول ٦-٦: طرق كائن Workspace

الطريقة	الشرح
BeginTrans	تبدأ تعاملاً جديداً
CommitTrans	تنهي التعامل وتحفظ التغييرات
Rollback	تنهي التعامل وتتراجع عن جميع التغييرات
Close	تنهي جلسة العمل
CreateDatabase	تنشئ وتحفظ قاعدة بيانات خالية جديدة
CreateGroup	تنشئ كائن Group جديد
CreateUser	تنشئ كائن User جديد
OpenDatabase	تفتح قاعدة بيانات واحدة أو اكثر في مسافة عمل Jet
OpenConnection	تفتح مصدر بيانات من نوع ODBC في مسافة ODBCDirect تستخدم للاتصال المباشر بملقم ODBC بدون تحميل محرك قاعدة البيانات Jet

كائن قاعدة البيانات "Database"

يمثل كائن Database قاعدة بيانات مفتوحة. يمكن فتح أكثر من قاعدة بيانات في وقت واحد في مسافة العمل، ويمكن فتح قاعدة بيانات موجودة بالفعل في إجراء VBA باستخدام طريقة OpenDatabase لكائن Workspace. وبطبيعة الحال، عندما يكون هناك أكثر من قاعدة بيانات مفتوحة، فإنك تشير إلى اسم كل واحدة على حدة كما هو موضح في المثال التالي:

```
DBEngine(0)![Northwind.mdb]
DBEngine(0).Databases("Northwind.mdb")
```

للإشارة إلى قاعدة البيانات الحالية، يمكن استخدام وظيفة ()CurrentDB في أكسس، يوضح الجدولان ٦-٧ و٦-٨ خصائص وطرق كائن Database.

الجدول ٦-٧: خصائص كائن Database

الخاصية	التشغيل	نوع البيانات	الشرح
CollatingOrder	Read-Only	سلسلة	تحدد الترتيب التصنيفي لمقارنات السلسلة أو تصنيفها
Connect	Read-Write	سلسلة	تقدم معلومات عن مصدر قاعدة البيانات المفتوح، متضمناً نوع قاعدة البيانات ومسارها
Connection	Read-only	كائن	تعيد كائن Connection الذي يطابق قاعدة البيانات يستخدم في تغيير الاتصالات بمصدر بيانات من نوع ODBC
Name	Read-only	سلسلة تتكون من أربعة وستين حرفاً كحد أقصى	تحدد أو تعيد الاسم المعرف من قبل المستخدم لكائن Database

الجدول ٦-٧: خصائص كائن Database

الخاصية	التشغيل	نوع البيانات	الشرح
QueryTimeout	Read/write	عدد صحيح	يعد أو يعيد عدد الثواني المفترض أن ينتظرها Jet قبل ظهور خطأ يتم عن انقضاء الوقت وذلك عند تشغيل استعلام على قاعدة بيانات ODBC.
RecordsAffected	Read-only	Long	يعيد عدد السجلات التي تأثرت بعد تشغيل إجراء استعلام
ReplicaID	Read-only	GUID	يعيد رقم تحديد الهوية المنفرد الذي يتكون من ستة عشر بايت، والذي أنشأه Jetعند عمل. قاعدة البيانات
Transactions	Read-only	منطقي Boolean	يحدد ما إذا كانت قاعدة البيانات تدعم التعامل باعتباره سلسلة من التغييرات التي يمكن حذفها أو حفظها فيما بعد
Updatable	Read-only	Boolean	يحدد ما إذا كانت التغييرات يمكن أن تحدث لكائن دخول البيانات

الجدول ٦-٧: خصائص كائن Database

الخاصية	التشغيل	نوع البيانات	الشرح
Version	Read-only	سلسلة String	يعيد إصدار محرك قاعدة البيانات في Jet الذي أنشأ ملف mdb، وفي مسافة يمل من نوع ODBCDirect يعيد إصدار برنامج ODBC المستخدم حالياً

الجدول ٦-٨: طرق كائن Database

الطريقة	الشرح
Close	يقوم بإغلاق كائن Database المفتوح
CreateProperty	يقوم بإنشاء كائن Property لقاعدة البيانات معرف من قبل المستخدم
CreateQueryDef	تقوم بإنشاء استعلام جديد "كائن QueryDef" في قاعدة بيانات متعددة
CreateRelation	تقوم بإنشاء علاقة جديدة "كائن Relation" بين حقول جدولين أو استعلامين
CreateTableDef	تقوم بإنشاء جدول جديد "كائن TableDef". يجب أن يحتوي الجدول على حقل واحد على الأقل قبل أن تحفظ الجدول "وذلك عن طريق إضافة كائن TableDef إلى مجموعة TableDef"
Execute	تقوم بتشغيل إجراء استعلام "إجراء مخزن أو عبارة SQL"
MakeReplica	تقوم بعمل نسخة جديدة من قاعدة بيانات مستندة إلى قاعدة البيانات الحالية
NewPassword	تقوم بتغيير كلمة المرور لقاعدة بيانات Jet

الجدول ٦-٨: طرق كائن Database

الطريقة	الشروح
OpenRecordset	تقوم بإنشاء كائن Recordset جديد وإضافة إلى مجموعة Recordsets
PopulatePartial	تزامن التغييرات في نسخة مماثلة جزئية مع النسخة المماثلة الكلية
Synchronize	تزامن نسختين متماثلتين

كائن TableDef

يمثل كائن TableDef تعريف الجدول المخزن. يمكن للجدول أن يستخدم قاعدة بيانات Jet (يعرف باسم جدول محلي أو جدول قاعدة) أو في قاعدة بيانات أخرى متعلقة بقاعدة بيانات Jet. كائنات تشغيل البيانات الوحيدة التي تمثل البيانات المخزنة في الجدول هي كائنات Field في مجموعة Field لكائن مجموعة السجلات (Recordset). يوضح الجدولان "٦-٩" و"٦-١٠" خصائص وطرق كائن TableDef.

الجدول ٦-٩: خصائص كائن TableDef

الخاصية	التشغيل	نوع البيانات	الشرح
Attributes	Read/write	Long	تحدد خصائص الجدول فتحدد مثلاً ما إذا كان الجدول جدولاً مرتبطاً أو جدولاً محلياً.
ConflictTable	Read-only	String سلسلة	يعيد اسم الجدول الذي يحتوي على السجلات التي تعارضت مع بعضها البعض أثناء تزامن نسختين متماثلتين.
Connect	Read/write	String سلسلة	يقدم معلومات عن مصدر الجدول متضمناً نوع قاعدة البيانات ومسارها.

الجدول ٦-٩: خصائص كائن TableDef

الخاصية	التشغيل	نوع البيانات	الشرح
DateCreated	Read-only	Variant	يعيد تاريخ ووقت إنشاء الجدول.
LastUpdated	Read-only	Variant	يعيد تاريخ ووقت أخر تغيير تم إحداثه بالجدول.
Name	Read/write	سلسلة تتكون من أربعة وستين حرفاً على الأكثر	يعد أو يعيد اسم الجدول المعرف من قبل المستخدم.
RecordCount	Read-only	Long	يعيد رقم السجلات في الجدول.
ReplicaFilter	Read/write	String or Boolean	يحددها إذا كانت المجموعة الجزئية للسجلات قد تم عمل نسخة مماثلة لها في الجدول من نسخة أخرى مماثلة مكتملة.
SourceTableName	Read/writeforAttachedtable,readonlyforLocaltable	String	تحدد اسم الجدول المرتبط أو الجدول المحلي.
Updatable	Read-only	Boolean	تحدد ما إذا كان يمكن عمل تغييرات لتعريف الجدول.
ValidationRule	Read/write	String	تجعل من البيانات في الحقل أو في مجموعة الحقول بيانات صالحة وذلك عند تغيير البيانات أو إضافتها إلى الجدول.
ValidationText	Read/write	String	تحدد نص الرسالة التي تعرض حالة عدم إرضاء إعداد ValidationRule.

الجدول ٦-١٠: طرق كائن TableDef

الطريقة	الشرح
CreateField	تقوم بإنشاء كائن Field جديد للجدول.
CreateIndex	تقوم بإنشاء فهرس جديد "كائن Index" للجدول. يجب أن يكون للفهرس حقل واحد على الأقل.
CreateProperty	تقوم بإنشاء خاصية جديدة معرفة من قبل المستخدم "كائن Property" لجدول موجود بالفعل.
OpenRecordset	تقوم بإنشاء كائن Recorset بالجدول وإضافة إلى مجموعة Recprset.
RefreshLink	تقوم بتحديث المعلومات عن الاتصال لجدول مرتبط بعد إعادة تعيين خاصية Connect بالجدول.

كائن Field

يمثل كائن Field عموداً من البيانات لها مجموعة عامة من الخصائص ونوع عام من البيانات وتحتوي جميع كائنات TableDef وQueryDef وIndex وRelation على مجموعات Fields تحتوي بدورها على كائنات Field. في هذه الحالات، تتضمن خصائص كائن Field مواصفات الحقل لكنها لا تحتوي على بيانات. مثل هذه الكائنات لا يكون لها خاصية Value. يمكن استخدام نافذة Immediate لتقييم خصائص Field:

- اكتب؟ CurrentDB!Customers!CustomerID.AllowZeroLength ثم اضغط Enter لعرض إعداد الخاصية.
- اكتب؟ CurrentDB!Customers!CustomerID.Value ثم اضغط Enter. سيعرض أكسس رسالة خطأ نظراً لأن كائن Field لكائن TableDef لا يحتوي على أية معلومات، ومن ثم لا يكون له خاصية Value.

إذن فأين تكون البيانات؟ إنها في حقول مجموعة السجلات التي يقوم Jet بإنشائها عند فتح جدول أو تشغيل استعلام أو عبارة SQL التي تعيد السجلات. يحتوي كائن Recordset على مجموعة Fields تحتوي على كائنات Field. في مثل هذه الحالة، تحتوي مجموعة Fields على حقول تمثل صفاً واحداً من البيانات الحقيقية، أي البيانات الموجودة في السجل الحالي. استخدم كائنات Field في كائن Recordset لاختبار أو تغيير البيانات في السجل الحالي. تعيد خاصية

Value لكائن Field قيمة البيانات في حقل من حقول السجل الحـــالي. يمكـــن اســـتخدام نـــافذة Immediate لتقييم خصائص كائن Field:

- اكتب؟ rstCustomers.CustomerID.Value ثـم اضغـط Enter لـعرض إعـداد الخاصية.
- اكتب؟ rstCustomers.CustomerID.Value ثم اضغط Enter لعرض قيمة الحقـل. أو، نظراً لأن Value هي الخاصية الافتراضية في هذه الحالة، فإنه يمكن عرض قيمـة البيانات في الحقل عن طريق كتابة؟ rstCustomers.CustomerID ثم ضغط Enter.

تعتمد خصائص وطرق كائن Field على الكائن الأساسي لها، فالكائن الأساسي هو ذلك الـذي يحتوي على مجموعة Fields التي يضاف إليها كائن Field: Index (I), QueryDef (Q), Recordset (Rec), Relation (Rel), and TableDef (T) يوضح الجدولان "٦-١١" و"٦-١٢ خصائص وطرق كائن Field "عن طريق استخدام اختصارات الكائن الأساسي".

الجدول ٦-١١: خصائص كائن Field

الخاصية	التشغيل	نوع البيانات	الشرح
AllowZeroLength	Read/Write for T;read-only for Q and Rec	Boolean منطقي	يحدد ما إذا كانت السلسلة التي يقدر طولها بصفر إعداداً صالحاً لخاصية Value لحقل له بيانات من نوع Text أو Memo.
Attributes	Read/Write for T;read-only for Q and Rec read-only for I after appending T	Long	يحدد خصائص الحقل، فيحدد مثلاً ما إذا كان حجم الحقل ثابت أم متغير.
CollatingOrder	Read only for Q, Rec, and T	Long	تحدد الترتيب التصنيفي لمقارنات السلسلة أو تصنيفها.

الجدول ٦-١١: خصائص كائن Field

الخاصية	التشغيل	نوع البيانات	الشرح
DataUpdatable	for Q, Rec, and Rel	Boolean	تحدد ما إذا كان يمكن تغير خاصية Value للحقل.
DefaultValue	Read/Write for T;read-only for Q and Rec	String	تعد أو تعيد القيمة التي يتم إدخالها تلقائياً على أنها قيمة الحقل وذلك عند إنشاء سجل جديد. لا يكون لكائنات Field لكائنات Index وRelation خاصية DefultValue.
FieldSize	Read only for Rec	Long	تعيد عدد الأحرف الخاص بحقل Memo، أو عدد وحدات البايت لحقل Long Binary ومع أنواع البيانات الأخرى، استخدام خاصية Size.
ForeignName	Read/Write for Rel; read-only after appending	String	تعد أو تعيد قيمة تحدد اسم الحقل في جدول غريب أو استعلام يكون مطابقاً لحقل من الحقول الموجودة في الجدول الأساسي أو الاستعلام.
Name	Read/Write للحقل الجديد، read only بعد إضافة "إلا في حالة الجدول المحلي"	سلسلة تتكون من أربعة وستين حرفاً على الأكثر	تعد أو تعيد القيمة الاسم.

الجدول ٦-١١: خصائص كائن Field

الخاصية	التشغيل	نوع البيانات	الشرح
OrdinalPosition	Read/Write for T;read-only for Q and Rec	Integer	تعد أو تعيد الموضع النسبي في مجموعة Field التي تكون مبنية على الصفر. وإذا اشترك حقلين أو أكثر نفس قيمة OrdinalPosition فإنه يتم ترتيبها أبجدياً.
Required	Read/Write for T;read-only for Q and Rec	Boolean	يحدد ما إذا كان الحقل يتطلب قيمة غير فارغة.
Size	Read/Write for T, Q, and Rec; read only after appinding	Long	يعيد أكبر حجم للحقل بوحدة البايت.
SourceField	Read-only for Q, Rec, and T	String	يعيد اسم الحقل الذي يعد المصدر الأصلي لبيانات الحقل، فعلى سبيل المثال، يمكن تحديد اسم حقل الجدول الذي يمد حقل الاستعلام بالبيانات. يتوفر فقط في وقت التشغيل.
SourceTable	Read-only for Q, Rec, and T	String	يعيد اسم الجدول الذي يعد المصر الأصلي لبيانات الحقل. يتوفر فقط وقت التشغيل.

الجدول ٦-١١: خصائص كائن Field

الخاصية	التشغيل	نوع البيانات	الشرح
Type	Read/Write read only after appinding	Integer	يحدد نوع بيانات الحقل.
ValidateOnSet	Read/Write for Rec	Boolean	يحدد ما إذا كانت قيمة كائن Field صالحة عند إعداد خاصية Value للكائن (True) أو عند تحديث السجل (False).
ValidationRule	Read/Write for T (Local Table); read-only for Q, Rec, and T (linked table)	String	يجعل من البيانات بيانات صالحة في حقل عند تغيير البيانات أو إضافتها إلى الجدول.
ValidationText	Read/Write for T; read only for Q, and Rec	String	يحدد نص الرسالة الذي تعرض في حالة عدم إرضاء إعداد ValidationRule.
VisibleValue	Read-only	Variant	تعيد القيمة التي توجد في الوقت الحالي في الحقل الموجود بقاعدة البيانات على الملقم "فقط مسافات عمل ODBC Direct".
Value	Read/Write for Rec	Variant	تسترد أو تغير البيانات في كائنات Recordset.

الجدول ٦-١٢: طرق كائن Field

الطريقة	الشـرح
AppendChunk	تضيف بيانات من تعبير سلسلة لكائن Memo أو كائن OLE Object Field. يمكن تطبيقها على مجموعة السجلات (Recorset).
CreateProperty	تقوم بإنشاء خاصية Property جديدة معرفة من قبل المستخدم، لكائن مخزن بالفعل على قرص. يمكن تطبيقها على كائنات QueryDef, Recordset, Relation, and TableDef.
GetChunk	تعيد جميع محتويات كائن Memo أو كائن OLE Object Field أو جزء منها (متغير سلسلة أو متغير) يمكن تطبيقها على كائن Recordset.

كائن Index

يمثل كائن Index فهرس لجدول قاعدة البيانات. يستخدم الفهرس لغرضين أساسيين: ترتيب السجلات التي تم إعادتها في كائن Recordset من نوع table يستند إلى جدول مخزن، ولتحديد ما إذا كان للسجلات قيماً مكررة في الحقول يستعاض بها عن الفهرس لن تحتاج إلى إنشاء فهرس لجدول مخزن، لكن نظراً لأن Jet قادرة على تحديد مواضع السجلات وإنشاء واصلات بكفاءة أعلى، باستخدام الفهارس، فإنك بطبيعة الحال تقوم بإنشاء فهارس عديدة بكل جدول. ويصون Jet جميع فهارس الجداول في قاعدة البيانات، فيقوم بتحديث الفهارس تلقائياً عند تعديل أو إضافة أو حذف سجلات في الجدول.

لكن هناك حالة واحدة، يكون مطلوب فيها إنشاء فهرس إذا أردت إنشاء علاقة بين جدولين، يجب أولاً تعريف مفتاح أساسي للجدول على أحد الجوانب لإحدى العلاقات المتعددة، وسيكون الحقل المطابق "أو الحقول" في الجدول والموجود بالجوانب المتعددة للعلاقة هو المفتاح الغريب. يقوم Jet تلقائياً بتعريف الفهرس الأساسي للجدول "الأوحد" ليكون المفتاح الأساسي لهذا الجدول. بالإضافة إلى ذلك، في حالة فرض العلاقة لتكامل مرجعي، يقوم Jet تلقائياً بإنشاء فهرس أخر للجدول "الأوحد"، وذلك مع إعداد خاصية Foreign في المفتاح الغريب للجدول "المتعدد".



ملاحظة: تخزن سجلات جدول قاعدة البيانات في الصفحات. إذا أردت إيجاد سجلات ذات قيمة معينة، مثل سجلات عن جميع العملاء في الأرجنتين، يقوم أكسس بالبحث في جميع الصفحات عن جدول Customers لإيجاد السجلات. والفهرس هو جدول بحث (مخزن منفصل في مجموعة الصفحات الخاصة به) يجعل قيمة الحقل في جدول البحث مرتبطة بموضع الصفحة التي تحتوي على سجل له قيمة حقل في الفهرس، تكون القيم مرتبة ترتيباً تصاعدياً. عند استخدام الفهرس في البحث، يقوم أكسس بقراءة صفحات الفهرس وإيجاد القيمة المفهرسة ثم البحث عن صفحة أو صفحات البيانات التي تحتوي على السجل أو السجلات المطابقة. ونظراً لأن القيم في الفهرس تكون مرتبة، وفي نفس الوقت يكون الفهرس صغيراً (متضمناً فقط حقول الفهرس لا جميع الحقول التي وردت بالجدول)، فإن أكسس يمكنه إيجاد القيمة بسرعة.

عند إنشاء فهرس في واجهة أكسس، يهتم أكسس تلقائياً بالتفاصيل، لكن عند إنشاء فهرس في إجراء VBA، فإنه يجب عليك أن تخطو في كل عملية بوضوح وبالترتيب. فعلى سبيل المثال، في Northwind يمكن إنشاء فهرس لجدول Employees يحتوي على الحقول LastName وFirstName وها هي الخطوات التي يجب اتباعها في إجراء VBA عند إنشاء هذا الفهرس "انظر الفصل الرابع عشر لمزيد من التفاصيل عن إنشاء إجراء VBA للفهرس".

١- قم بإنشاء وتسمية الفهرس وذلك بتحديد خاصية Name للفهرس، مثل خاصية FullName.

٢- قم بإنشاء كل حقل في الفهرس وقم بإعداد خاصية Name للحقل في اسم الحقل. في هذه الحالة قم بإنشاء حقلين: FirstName وlastName.

٣- أضف كل حقل إلى مجموعة Fields للفهرس.

٤- أضف الفهرس إلى مجموعة Indexes للجدول.

يوضح الجدول ٦-١٣ خصائص كائن Index، الذي له طريقان:

Create Field لإنشاء حقل جديد للفهرس، وcreate Property لإنشاء خاصية جديدة للفهرس معرفة من قبل المستخدم.

الجدول ٦-١٣: خصائص كائن Index

الخاصية	التشغيل	نوع البيانات	الشرح
Clustered	Read/Write; Read-only بعد الإضافة	Boolean	يحدد ما إذا كان كائن الفهرس يمثل فهرساً متشابكاً للجدول.
DistinctCount	Read-only	Long	يحدد عدد القيم أو المفاتيح في الفهرس.
Foreign	Read-only	Boolean	يحدد ما إذا كان كائن Index يمثل مفتاحاً أساسياً في الجدول. يقوم Jet بإعداد خاصية Foreign عند إنشاء علاقة تكامل مرجعي.
IgnoreNulls	Read/Write; Read-only after appending	Boolean	تحدد ما إذا كانت السجلات التي لها قيم Null في حقول الفهرس متضمنة في الفهرس، لتصغير حجم الفهرس، قم بإعداده بحيث يكون True لاستبعاد هذه السجلات.
Name	Read/Write لفهرس جديد، Read-only للفهرس الموجود	سلسلة تتكون من أربعة وستين حرف على الأكثر	يعد أو يعيد اسم الفهرس المعرف منة قبل المستخدم.

الجدول ٦-١٣: خصائص كائن Index

الخاصية	التشغيل	نوع البيانات	الشرح
Primary	Read/Write; Read-only بعد الإضافة	Boolean	يحدد ما إذا كان كائن الفهرس يمثل فهرساً أساسياً للجدول.
Required	Read/Write; Read-only بعد الإضافة	Boolean منطقي	يحدد ما إذا كان يجب تعبئة جميع الحقول في كائن Index.
Unique	Read/Write; Read-only بعد الإضافة	Boolean منطقي	تحدد ما إذا كان كائن Index يمثل فهرساً واحداً للجدول.

كائن Relation

يمثل كائن Relation علاقة بين حقول في جداول أو في استعلامات. يستخدم كائن Relation في إجراء VBA لإنشاء علاقة جديدة أو لتعديل أو لفحص لعلاقة موجودة بالفعل. تربط العلاقة أحد صفوف الجدول أو الاستعلام اللذان يعرفان باسم الجدول الأساسي أو الاستعلام الأساسي، تربطه بأي عدد من الصفوف في جدول أو استعلام أخر يعرف باسم الجدول الغريب أو الاستعلام الغريب. يمكن إنشاء علاقة بين جدولين، أو بين جدول واستعلام أو بين استعلامين، وقد تكون الجداول محلية أو مرتبطة. على الرغم من أن الاستعلام والجداول المرتبطة قد تكون عناصر للعلاقة، فإن Jet لا يستطيع فرض التكامل المرجعي على العلاقة إلا إذا كانت هذه العناصر جداول محلية.

يمكن إنشاء العلاقة عن طريق تحديد حقل مطابقاً "أو حقول" في كل عنصر. يجب أن يكون الحقل أو الحقول المطابقة في الجدول الأساسي أو في الاستعلام الأساسي، يجب أن تكون مفتاحاً أساسياً يحدد صفاً ما. يعرف الحقل المطابق "أو الحقول" في الجدول الغريب أو الاستعلام الغريب باسم المفتاح الغريب. لإنشاء علاقة أضف كل حقل مفتاح أساسي إلى مجموعة Fields للعلاقة وحدد اسم الحقل المطابق في المفتاح الغريب مستخدماً خاصية ForeignName.

عند إنشاء علاقة في وجهة أكسس، يعتني أكسس بالتفاصيل تلقائياً، غير أنه عند إنشاء علاقة في إجراء VBA، فإنه يجب أن تخطو كل عملية وبالترتيب. فعلى سبيل المثال، في Northwind يمكن إنشاء علاقة بين جدول Customers باعتباره الجدول الأساسي الذي يحتوي على حقل المفتاح الأساسي Customer ID وبين جدول Orders باعتباره جدولاً غريباً يحتوي على حقل المفتاح الغريب Customer ID هاهي الخطوات التي يجب اتخاذها في VBA عند إنشاء هذه العلاقة. "انظر الفصل الرابع عشر للتعرف على مزيد من التفاصيل عن إنشاء إجراء VBA لعلاقة مماثلة":

١- قم بإنشاء وتسمية العلاقة بتحديد خاصية Name للعلاقة "مثل خاصية CistomersOrdersRelation" وخاصية Table باعتبارها اسم الجدول الأساسي أو الاستعلام وخاصية Foreign Table باعتبارها اسم الجدول الغريب أو الاستعلام.

٢- قم بإنشاء كل حقل للعلاقة، وذلك بتحديد حقل المفتاح الأساسي. في هذه الحالة، قم بإنشاء حقلاً واحداً للعلاقة وبإعداد خاصية الحقل Name في اسم حقل المفتاح الأساسي الوحيد Customer ID.

٣- حدد لكل حقل في العلاقة ForeignName باعتبارها اسم حقل المفتاح الغريب المطابق. في هذه الحالة، قم بإعداد خاصية ForeignName في Customer ID "في هذا المثال، يكون لحقلي المفتاح الأساسي والغريب نفس الاسم، لكنها قد تختلف في بعض الأحيان".

٤- أضف الحقل إلى مجموعة Fields لكائن Relation.

٥- أضف العلاقة إلى مجموعة Relations لقاعدة البيانات.

تستخدم خاصية Attributes لتحديد ما إذا كانت العلاقة بين حقل وأخر أو بين حقل وعدة حقول، أو إذا كانت العلاقة بين الحقول تتصل من اليمين أم من اليسار كما تستخدم خاصية Attributes لتحديدها إذا كنت تريد من Jet فرض التكامل المرجعي وتتالي الخيارات للعلاقة.

ملاحظة يمكن أن تعرض Jet التكامل المرجعي فقط للعلاقة بين جداول في قاعدة البيانات الحالية لكن قبل فرض التكامل المرجعي، يجب أن يكون هناك فهرس واحد للحقل المطابق في الجدول على أحد جوانب العلاقة، وعند فرض التكامل المرجعي، يقوم Jet تلقائياً بإنشاء فهرس للحقل المطابق في الجدول على الجوانب العديدة للعلاقة.

يوضح الجدول "٦-١٤" خصائص كائن Relation، وهذا الكائن يكون له طريقة واحدة، ألا وهي طريقة CreateField التي تستخدم لإنشاء حقل جديد للعلاقة.

الجدول ٦-١٤: خصائص كائن Relation

الخاصية	التشغيل	نوع البيانات	الشروح
Attributes	Read/Write	Long	يحدد خصائص العلاقة على أنها مجموع الثوابت الحقيقية: dbRelationUnique لعلاقة بين حقل وأخر، وdbRelationDontEnforce لغير التكامل المرجعي، وdbRelationInheriyed لعلاقة بين جدولين متعلقين ببعضهما البعض في قاعدة بيانات غير حالية، وdbRelationUpdateCascade لتتالي التحديثات، وdbRelationDeleteCascade لتتالي محذوفات السجل.
Foreign Table	Read/Write	String	يحدد أو يعيد اسم الجدول الغريب أو الاستعلام في العلاقة.
Name	Read/Write for new relation; read-only for existing one	String, maximum of 64 characters	يعد أو يعيد الاسم المعرف من قبل المستخدم للعلاقة.
Partial Replica	Read/Write	Boolean	يقوم بعمل نسخة مماثلة من البيانات من النسخة الكاملة في نسخة جزئية تستند إلى العلاقة بين الجداول.
Table	Read/Write	String	يعد أو يعيد اسم الجدول الأساسي أو الاستعلام في العلاقة.

كائن Recordset

يمثل كائن Recordset مجموعة السجلات الموجودة في قاعدة البيانات، أو يمثل مجموعة السجلات التي تنتج عن تشغيل استعلام أو عبارة SQL. وفي كائن Recordset الذي يحتوي على سجل واحد على الأقل، قم بالإشارة إلى سجل واحد في كل مرة باعتباره السجل الحالي. تحتوي مجموعة Fields لكائن Recordset على كائنات Fields التي تمثل الحقول في السجل الحالي، على وجه الخصوص، يكون إعداد خاصية Value لكل كائن Field هو قيمة البيانات المخزنة في الحقل في الجدول.

وتوضح المقاطع السابقة في هذا الفصل التقنيات التي تستخدم للتنقل في مجموعة السجلات ولمعالجة البيانات، وأنواع كائنات Recordset: table-type (TRec), dynaset-type (DRec), snapshot-type (SRec), and forward-only-type (FRec). يوضح الجدولان "٦-١٥" و"٦-١٦" خصائص وطرق كائن Recordset "باستخدام الاختصارات لنوع Recordset". لو لم يتم تحديد نوع معين، يتم تطبيق الخاصية أو الطريقة على الأنواع الأربعة لمجموعات السجلات.

ملاحظة تقدم مسافات العمل ODBC Direct نوعاً خاصاً لكائن Recordset يعرف باسم النوع الحيوي. تحتوي مجموعات السجلات في مسافات العمل ODBC على خصائص وطرق إضافية لا تتوافر في مسافة عمل Jet. انظر Access 2000 Developer's Handbook by Paul Litwin, Ken Getz, and Mike Gilbert (Sybex, 1999) لمزيد من المعلومات بهذا الصدد.

الجدول ٦-١٥: خصائص كائن Recordset من نوع DAO

الخاصية	التشغيل	نوع البيانات	الشرح
AbsolutePosition	Read/Write for DRec and Srec	Long	يوجه مؤشر السجل الحالي إلى سجل محدد مستند في ذلك إلى موضعه العادي، المستند إلى الرقم صفر.
BatchCollisionC	Read-only	Long	يعيد عدد السجلات التي لم

الجدول ٦-١٥: خصائص كائن Recordset من نوع DAO

الخاصية	التشغيل	نوع البيانات	الشرح
ount			ينجح تحديثها في أخر تحديث دفعي "فقط مسافات العمل ODBC Direct".
BatchCollisions	Read-only	Array of variants	يعيد مصفوفة من الإشارات المرجعية للسجلات التي لم ينجح تحديثها في أخر تحديث دفعي "فقط مسافات العمل ODBC Direct".
BatchSize	Read/write	Long	يعد أو يعيد عدد العبارات التي تم إرسالها إلى الملقم في تحديث دفعي واحد "فقط مسافات العمل ODBC Direct".
BOF, EOF	Read-only	Boolean	يحدد ما إذا كان كائن Recordset يحتوي على سجلات أو إذا كان عليك تخطي الحدود.
Bookmark	Read-only for TRec, DRec, and Srec		يحفظ مكانك في كائن Recordset.
Bookmarkable	Read-only for TRec, DRec, and Srec	Boolean	يحدد ما إذا كان كائن Recordset يدعم إشارات مرجعية.
CacheSize	Read/write	Long	يحدد عدد السجلات في كائن من نوع المجموعة الحيوية "Dynaset" الذي يحتوي على بيانات من مصدر ODBC يمكن تخزينها مؤقتاً محلياً.

الجدول ٦-١٥: خصائص كائن Recordset من نوع DAO

الخاصية	التشغيل	نوع البيانات	الشروح
CacheStart	Read/write	String	يعد أو يعيد الإشارة المرجعية الخاصة بالسجل الأول في كائن Recordset ليتم تخزينه تخزيناً مؤقتاً.
Connection	Read-only for DRec, Srec, and Frec	Object	يعيد متغير الكائن الذي يمثل الاتصال بمصدر بيانات ODBC لمسافة عمل ODBC Direct.
DateCreated	Read-only for TRec	Variant	يعيد تاريخ ووقت إنشاء مجموعة السجلات.
EditMode	Read-only	Integer	يحدد ما إذا كنتءقد استدعيت طريقة Edit أم طريقة Add New.
Filter	Read/Write for DRec, Srec, and Frec	String	تحدد السجلات التي يتضمنها كائن Recordset مفتوح.
Index	Read/Write for TRec	String	تحدد اسم كائن الفهرس الحالي باعتباره الفهرس الحالي لكائن Recordset من نوع Table.
LastModified	Read-only for TRec, DRec, and Srec		تعيد إشارة مرجعية تحدد أخر سجل تم تغييره أو إضافته، تستخدم للانتقال إلى هذا السجل.
LastUpdated	Read-only for TRec	Variant	تعيد تاريخ ووقت أخر مرة تم تغيير مجموعة السجلات فيها.
LockEdits	Read/Write for TRec, DRec, and Srec	Boolean	تحدد نوع التأمين الذي يستخدم أثناء التخزين متشائم "True" أو متفائل "False".

الجدول ٦-١٥: خصائص كائن Recordset من نوع DAO

الخاصية	التشغيل	نوع البيانات	الشرح
Name	Read-only	String	تعيد اسماً لكائن Recordset.
NoMatch	Read-only for TRec, DRec, and Srec	Boolean	تحدد إذا ما تم إيجاد سجل معين باستخدام إحدى طرق Find أو طريقة Seek. تعيد القيمة False في حالة إيجاد السجل.
PercentPosition	Read/Write for TRec, DRec, and Srec	Single	يحدد الموضع التقريبي للسجل الحالي مستنداً إلى نسبة مئوية من السجلات في مجموعة السجلات.
RecordCount	Read-only	Long	تعيد عدد السجلات في كائن Recordset من نوع Table، أو تعيد عدد السجلات التي يتم تشغيلها في كائن Dynaset أو Snapshot.
RecordStatus	Read-only	Long	تعيد قيمة توضح حالة التحديث للسجل الحالي في تحديث دفعي "فقط مسافات العمل ODBC Direct".
Restartable	Read-only	Boolean	تعيد قيمة توضح ما إذا كان كائن Recordset يدعم طريقة Requery.
Sort	Read/Write for DRec and SRec	String	يحدد عبارة ORDER BY لعبارة SQL بدون عبارة ORDER BY. بعد إعداد خاصية Sort، يظهر التصنيف عند إنشاء كائن Recordset متتالي من الكائن.

الجدول ٦-١٥: خصائص كائن Recordset من نوع DAO

الخاصية	التشغيل	نوع البيانات	الشروح
StillExecuting	Read-only	Boolean	تحدد ما إذا كانت العملية التي تم تنفيذها على أنها طريقة لا متزامنة قد انتهت أم لا "فقط مسافات عمل ODBC Direct".
Transactions	Read-only	Boolean	تحدد ما إذا كانت مجموعة السجلات تدعم الإجراءات.
Type	Read-only	String	تعيد قيمة لتحديد ما إذا كان كائن Recordset من نوع Table أم Dynaset أم Snapshot أم Forward-only.
Updatable	Read-only	Boolean	تعيد قيمة لتحديد ما إذا كان يمكن إحداث تغييرات لكائن Recordset.
UpdateOptions	Read/write	Long	تعد أو تعيد قيمة لتحديد كيف تم بناء عبارة WHERE لكل سجل أثناء التحديث الدفعي "فقط مسافات عمل ODBC Direct".
ValidationRule	Read-only	String	تجعل البيانات بيانات صالحة في حقل عند تغييرها أو إضافتها إلى جدول.
ValidationText	Read-only	String	تحدد نص الرسالة التي تعرض في حالة عدم إرضاء قيمة كائن Field لإعداد خاصية ValidationRule.

الجدول ٦-١٦: طرق كائن Recordset

الطريقة	الشرح
Add New	تقوم بإنشاء سجل جديد في النسخة المؤقتة لكائن من نوع Table أو Dynaset. والسجل الذي كان هو السجل الحالي قبل استخدام طريقة Add New يظل سجلاً حالياً.
Cancel	تقوم بإلغاء تنفيذ استدعاء طريقة اللاتزامن "فقط لمسافات عمل ODBC Direct".
CancelUpdate	تقوم بإلغاء أي تحديثات تم إضافتها تبعاً لعملية Edit أو Add New.
Clone	تقوم بإنشاء كائن Recordset مكرر يشير إلى الكائن الأصلي، وفي البداية ينقص هذه النسخة سجلاً حالياً "لا تطبق على الكائنات من نوع Forward-only".
Close	تقوم بإغلاق كائن Recordset المفتوح "لا يؤثر على النسخة".
CopyQueryDef	تقوم بإنشاء نسخة من QueryDef تستخدم لإنشاء كائن Recordset.
Delete	تزيل السجل الحالي من كائن Table أو Dynaset مفتوح.
Edit	ينسخ السجل الحالي الموجود في كائن Table أو Dynaset ويضعها في النسخة المؤقتة لتكون جاهزة للتحرير.
FillCache	تقوم بتعبئة المخزن المؤقت بوضوح، الذي أنشأه Jet في حالة احتواء كائن Dynaset على بيانات من مصدر بيانات ODBC.
FindFirst, FindLast, FindNext, FindPrevious	تحدد موضع السجل الأول والأخير والتالي والسابق في كائن Dynaset أو Snapshot الذي يكون كافياً للمعيار المحدد، ويجعل من هذا السجل سجلاً حالياً. المعيار هو تعبير سلسلة لعبارة WHERE في عبارة SQL بدون كلمة WHERE.
Get Rows	تنقل سجلاً واحداً أو جميع السجلات من مجموعة السجلات إلى مصفوفة ثنائية الأبعاد، وتنقل مؤشر السجل الحالي إلى الصف التالي الذي لم يقرأ.

الجدول ٦-١٦: طرق كائن Recordset

الطريقة	الشرح
Move	تنقل عدداً محدداً من الصفوف إلى سجل أخر وتجعل منه سجلاً حالياً.
MoveFirst, MoveLast, MoveNext, MovePrevious	تنقل إلى السجل الأول أو الأخير أو التالي أو السابق في كائن Recordset، ويجعل من هذا السجل سجلاً حالياً "في نوع Forward-only لا تتوافر سوى طريقة MoveNext".
Next Recordset	تنتقل إلى مجموعة السجلات التالية، لو كانت توجد مجموعة بالفعل قد تم إعادتها عن طريق عبارة OpenRecordset التي تحتوي على أكثر من عبارة Select "فقط لمسافات عمل ODBC Direct".
Open Recordset	تقوم بإنشاء كائن Recordset جديد إذا كان الكائن الأصلي من نوع Table، يكون الكائن الجديد من نوع Dynaset، وإذا كان الكائن الأصلي من نوع Dynaset أو Snapshot يكون الكائن الجديد من نفس نوع الكائن الأصلي.
Requery	تقوم بتحديث البيانات في كائن Dynaset أو Snapshot أو Forward-only عن طريق إعادة تنفيذ الاستعلام الذي يستند إليه الكائن عند استخدام هذه الطريقة، يصبح السجل الأول هو السجل الحالي.
Seek	في كائن Table، تحدد هذه الطريقة موضع السجل في كائن Table مفهرس، وتكون كافية للمعيار المحدد للفهرس الحالي، كما تجعل من السجل الذي تم إيجاده سجلاً حالياً يجب تعيين خاصية Index في كائن Index موجود بالفعل قبل استخدام هذه الطريقة.
Update	تحفظ محتويات النسخة المؤقتة لكائن Dynaset أو Table محدد. في حالة عدم استخدام طريقة Update بوضوح، تضيع جميع التغييرات التي تم إحداثها بالسجل.

كائن QueryDef

يمثل كائن QueryDef التعريف المخزن للاستعلام، لكنه لا يمثل البيانات المخزنة في الجداول. يوضح الجدولان "٦-١٧" و"٦-١٨" خصائص وطرق كائن QueryDef.

الجدول ٦-١٧: خصائص كائن QueryDef

الخاصية	التشغيل	نوع البيانات	الشرح
CacheSize	Read/Write	Long	تحدد عدد السجلات التي تم استردادها من مصدر بيانات ODBC والتي سوف تخزن في الذاكرة المحلية "في مخزن مؤقت".
Connect	Read/Write	String	تقدم معلومات عن مصدر الجدول المرفق.
DateCreate	Read-only	Variant	تعيد تاريخ ووقت إنشاء الاستعلام المخزن.
LastUpdate	Read-only	Variant	تعيد تاريخ ووقت المرة الأخيرة التي تم فيها تغيير الاستعلام المخزن.
MaxRecords	Read/Write	Long integer	تحدد أقصى عدد ممكن من السجلات التي سيتم إعادتها عن طريق استعلام يعيد البيانات من قاعدة بيانات ODBC.
Name	Read/Write; Read-only للاستعلام المؤقت	سلسلة تتكون من ٦٤ حرف على الأكثر	تحدد هوية كائن Querydef. يمكن إنشاء استعلام مؤقت عن طريق تعيين اسم Name في السلسلة التي يقدر طولها بصفر (" ").

الجدول ٦-١٧: خصائص كائن QueryDef

الخاصية	التشغيل	نوع البيانات	الشرح
ODBCTimeout	Read/Write	Integer	تحدد عدد الثواني التي يجب على Jet انتظارها قبل ظهور خطأ انقضاء الوقت عند تشغيل استعلام على قاعدة بيانات ODBC.
Perpare	Read/Write	Long	تحدد ما إذا كان الملقم ينفذ الاستعلام مباشرة أو ينشئ إجراء مخزناً مؤقتاً يستند إلى الاستعلام "فقط مسافات عمل ODBC Direct".
RecordsAffect-ed	Read-only	Long	تعيد عدد السجلات التي تأثرت في أخر مرة تم فيها تشغيل طريقة Execute على كائن QueryDef.
ReturnsRecor-ds	Read/Write	Boolean	تحدد ما إذا كان مرور عبارة SQL من خلال الاستعلام إلى قاعدة بيانات خارجية يعيد سجلات. يجب إعداد خاصية Connect قبل إعداد هذه الخاصية
SQL	Read/Write	String	تعد أو تعيد عبارة SQL التي تقوم بتعريف الاستعلام.
StillExecuting	Read/Write	Boolean	تحدد ما إذا كانت العملية التي تم تنفيذها على أنها طريقة لا تزامنية قد انتهت أم لا "فقط مسافات عمل ODBC Direct".
Type	Read-only	Integer	تحدد نوع الاستعلام.

الجدول ٦-١٧: خصائص كائن QueryDef

الخاصية	التشغيل	نوع البيانات	الشرح
Updatable	Read-only	Boolean	تحدد ما إذا كان يمكن إحداث تغييرات بالاستعلام.

الجدول ٦-١٨: طرق كائن QueryDef

الطريقة	الشرح
Cancel	تقوم بإلغاء طريقة اللاتزامن التي أضيفت "فقط مسافات عمل ODBC Direct".
Direction	تقوم بإغلاق كائن QueryDef.
Name	تقوم بإنشاء كائن Property جديد معرف من قبل المستخدم.
Type	تقوم بتشغيل استعلام إجراء أو بتنفيذ عبارة SQL لاستعلام إجراء على كائن QueryDef المحدد. ونظراً لأن استعلامات الإجراء لا تعيد أية سجلات، فإن طريقة Execute أيضاً لا تعيد أية سجلات.
Value	تقوم بإنشاء كائن Recordset جديد يستند إلى كائن QueryDef ثم تقوم بإضافته إلى مجموعة Recordsets.

كائن Parameter

يمثل كائن Parameter معامل استعلام في استعلام المعامل. بدلاً من أن تقوم بإنشاء كائنات Parameter ثم تضيفها إلى مجموعة Parameters، فإنك تقوم بإنشاء أكثر من معامل استعلام كجزء من تعريف استعلام المعامل. وبعد إنشاء استعلام المعمل، استخدم كائن Parameter للإشارة إلى معامل استعلام موجود بالفعل. يوضح الجدول "٦-١٩" خصائص كائن Parameter والذي ليس له أية طرق.

الجدول ٦-١٩: خصائص كائن Parameter

الخاصية	التشغيل	نوع البيانات	الشرح
Direction	Read/Write	Long	تحدد ما إذا كان كائن Parameter معامل إدخال أم إخراج أم الاثنين معاً "فقط مسافات عمل ODBC Direct".
Name	Read-only	String	تحدد اسماً للكائن.
Type	Read-only	Integer	تحدد نوع بيانات المعامل.
Value	Read/Write	Variant	تعد أو تعيد قيمة المعامل "خاصية افتراضية".

كائن Error

يوضح الجدول "٦-٢٠" خصائص كائن Error والذي ليس له أية طريق.

الجدول ٦-٢٠: خصائص كائن Error

الخاصية	التشغيل	نوع البيانات	الشرح
Description	Read-only	String	تعيد شرحاً "أو وصفاً" للخطأ.
Number	Read-only	Long	تعيد قيمة وقيمة تطابق الخطأ.
Source	Read-only	String	تعيد اسم الكائن أو التطبيق الذي تسبب في ظهور الخطأ.

كائن Property

يمثل كائن Property خاصية مضمنة أو خاصية لكائن تشغيل بيانات معرفة من قبل المستخدم.
تمتلك جميع كائنات تشغيل البيانات فيما عدا كائن Error "وكائن Connection في ODBC

"Direct مجموعة Properties تضم كائنات Property. يوضح الجـــدول "٦-٢١" خصـــائص كائن Property، والذي ليست له أية طرق.

الجدول ٦-٢١: خصائص كائن Property

الخاصية	التشغيل	نوع البيانات	الشرح
Inherited	Read-only	Boolean	تحدد ما إذا كانت الخاصية مأخوذة من كائن أخر.
Name	Read-only للخاصيـــة المضمنة أو الموجـــودة بالفعل، Read./Write لخاصية التخصيـــــص الجديدة	سلسلة تتكـــون من ٦٤ حـــرف على الأكثر	تقوم بتسمية الخاصية.
Type	Read/Write; read-only بعد الإضافة	Integer	تحدد نوع بيانات الخاصية
Value	Read/Write	Variant	تعـــد أو تحـــدد قيمـــة الخاصيـــة "خاصية افتراضية".

كائن Container

هناك العديد من التطبيقات الأخرى بجانب أكسس تستطيع إدارة بياناتها مثل مايكروسوفت إكســل وVisual Basic و Visual C++. عند العمل مع أحد هذه التطبيقـــات باعتبـــارك عميـــل COM يستخدم ملقم COM، فإنك تقوم بإنشاء كائنات تطبيق محددة. فعلى سبيل المثال، يستخدم تطبيـــق أكسس لإنشاء كائنات أكسس مثل النماذج والتقارير وصفحات تشغيل البيانات والماكرو والوحدات النمطية، في حين انه يستخدم إكسل في إنشاء كائنـــات إكســل مثـــل أوراق العمـــل والخرائـــط والوحدات النمطية. دائماً ما يقوم تطبيق العميل COM بتخزين الكائنات الخاصة به فـــي الملـــف الخاص به، فعلى سبيل المثال، يقوم إكسل بتخزين كائناته في ملفات xls. الخاصة به، في حيـــن ان أكسس يقوم بتخزين كائناته في ملف قاعدة البيانات mdb. الذي يديره Jet، فهو يتتبع كائنـــات أكسس مستخدماً مجموعة Containers التي يكون لها كائن Container منفصل لكل نوع مـــن أنواع كائنات أكسس. يوضح الجدول "٦-٢٢" خصـائص كائن Container، والذي ليس له أيـــة طرق.

الجدول ٦-٢٢: خصائص كائن Container

الخاصية	التشغيل	نوع البيانات	الشرح
AllPermissions	Read-only	Long	تعيد جميع الرخص التي يمكن تطبيقها على خاصية UserName الحالية لكائن Container.
Inherit	Read/Write	Boolean	تحدد ما إذا كان كائن Document الجديد الذي تم إنشاؤه في الحاوية له الرخص الافتراضية المُعدّة من أجل الحاوية.
Name	Read-only	String	تقوم بتسمية الكائن.
Owner	Read/Write	String	تحدد مالك الكائن واسم كائن User أو Group. ويتمتع مالك الكائن بميزات لا تتوفر لغيره.
Permissions	Read/Write	Long	تحدد الرخص التي يحصل عليها المستخدم الواحد أو مجموعة المستخدمين.
UserName	Read/Write	String	تحدد اسم المستخدم أو مجموعة المستخدمين أو اسم مالك الكائن.

كائن Document

يضم كائن Document معلومات عن مثال محدد للكائن مثل جدول محدد أو علاقة أو نموذج أو تقرير. ويوضح الجدول "٦-٢٣" خصائص كائن Document. ولكائن Document طريقة واحدة فقط، وهي طريقة CreateProperty التي تقوم بإنشاء خاصية تخصيص جديدة لكائن Document.

الجدول ٦-٢٣: خصائص كائن Document

الخاصية	التشغيل	نوع البيانات	الشرح
AllPermissions	Read-only	Long	تعيد جميع الرخص التي يمكن تطبيقها على خاصية UserName الحالية لكائن Document.
Container	Read-only	String	تعيد اسم كائن Container الذي يحتوي على كائن Document.
DateCreated	Read-only	Variant	تعيد تاريخ إنشاء الكائن.
LastUpdated	Read-only	Variant	تعيد تاريخ أخر مرة تم فيها تحديث الكائن.
Name	Read-only	String	تعيد الاسم الذي أعطاه المستخدم أو Jet للكائن.
Owner	Read/Write	String	توضح مالك الكائن.
Permissions	Read/Write	Long	تحدد الرخص التي يستخدمها المستخدم الواحد أو مجموعة المستخدمين.
UserName	Read/Write	String	تحدد اسم المستخدم أو مجموعة المستخدمين التي لها رخصة من أجل الكائن.

كائن User

يمثل كائن User مستخدماً محدداً تم السماح له باستخدام الكائنات في قاعدة البيانات عند تزويد مسافة العمل بالأمن. فقط حدد هوية "أو عرف" حساب المستخدم باسم المستخدم ومُعرّف شخصي. وعند تزويد عناصر الأمن المختلفة، يستطيع كل مستخدم الدخول مستخدماً اسم مستخدم وكلمة مرور، كما سيتمتع بميزات التشغيل التي تم إعدادها كرخص لمستخدمين محددين

ومجموعات مستخدمين محددة ويوضح الجدول "٦-٢٤" خصائص كائن User. ولكائن User طريقتان هما: طريقة Create Group التي تستخدم لإنشاء كائن Group جديد لكائن User، وطريقة New Password التي تستخدم لتغيير كلمة المرور الخاصة بأحد أرصدة المستخدمين الحالية.



يجب أن تكون على علم بكلمة المرور الحالية حتى تستطيع تغييرها، فإذا فقدت كلمة المرور، لن تتمكن أبداً من فتح قاعدة البيانات مرة أخرى.

الجدول ٦-٢٤: خصائص كائن User

الخاصية	التشغيل	نوع البيانات	الشرح
Name	Read/Write للمستخدم الجديد Read-only للمستخدم الحالي	سلسلة تتكون من ٢٠ حرف على الأكثر	تقوم بتسمية المستخدم.
Password	Write-only للكائنات الجديدة	سلسلة تتكون من ١٤ حرف على الاكثر.	تقوم بإعداد كلمة مرور لرصيد المستخدم. وقد تتضمن كلمات المرور أي حرف من أحرف ASCII فيما عدا الحرف صفر "0"، وتكون جميعها أحرف كبيرة.
PID	Write-only للكائنات الجديدة	سلسلة تتكون من أربعة إلى عشرين حرف هجائي عددي	تقوم بإعداد المعرف الشخصي لرصيد المستخدم أو مجموعة المستخدمين يستخدم Jet PID واسم الرصيد لتعريف المستخدم أو مجموعة المستخدمين لا يمكن إعادة PID لمستخدم حالي، وهي تكون دائماً أحرف كبيرة.

كائن Group

يمثل كائن Group مجموعة من أرصدة المستخدمين التي قمت بإعداد رخص تشغيل عامة لـها. يوضح الجدول ٦-٢٥ خصائص كائن Group. ولكائن Group طريقة واحـدة هـي طريقـة CreateUser التي تستخدم لإنشاء كائن User جديد للمجموعة.

الجدول ٦-٢٥: خصائص كائن Group

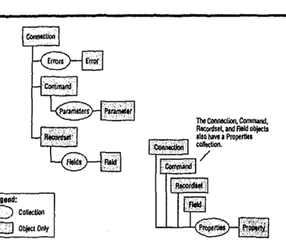
الخاصية	التشغيل	نوع البيانات	الشرح
Name	Read/Write للمجموعة الجديدة، read-only للمجموعة الحالية	سلسلة تتكـون من ٢٠ حـرف على الأكثر	تقوم بتسمية المجموعة.
PID	Write-only للمجموعة الجديدة	سلسلة تتكـون من ٤ إلـى ٢٠ حرف هجـائي عددي	تقـوم بـإعداد المعـرف الشـخصي للمجموعة الجديدة. لا يمكـن إعـادة PID لمجموعة قائمة.

تسلسل كائنات ActiveX Data الهرمي "ADO"

يعد تسلسل ADO الهرمي أبسط من مثيله DAO، حيث أن عدد كائنات ومجموعاته أقل. "انظـر الشكل ٦-٤". يأتي علي قمة هذا التسلسل كائن Connection الذي يمثل مفهوم الإجراء الـذي يحد بداية ونهاية سلسلة عمليات تشغيل البيانات التي توجد علي مدار الاتصال. ستلاحظ أن هناك بعض نقاط التشابه بين بعض أنـواع الكائنـات مثـل Recordsets وFields وParameters، فيكون لها أحيانا وظائف متشابهة. في حالة العمل في مشروع اكسس، يتم إدارة الكائنـات فـي تسلسل ADO الهرمي بواسطة MSDE.

الشكل ٦-٤

تسلسل كائنات ActiveX Data الهرمي.



والهدف من ADO هو تشغيل وتحرير وتحديث مصادر البيانات. ومع استخدام VBA، يمكن لمبرمج قاعدة البيانات إنشاء وحدات نمطية لتنفيذ تلك الوظائف.ويحتوي نموزج ADO على جميع الكائنات اللازمة من أجل:

- الاتصال بمصدر بيانات.
- إنشاء كائن يتضمن أمر SQL.
- تحديد أعمدة وجداول وقيم كمعامل متغير في أمر SQL.
- تنفيذ أمر SQL.
- تخزين نتائج الأمر في مخزن مؤقت.
- إنشاء تمثيل مرئي للمخزن المؤقت حتى يستطيع المستخدم تصنيف/تصفية ونقل البيانات.
- تحرير البيانات.
- تحديث مصدر البيانات بإضافة أي تغييرات أحدث في المخزن المؤقت إليه.
- قبول أو رفض التغييرات التي أحدث أثناء الإجراء، ثم إغلاق الإجراء.

ملاحظة لا يسمح نموذج ADO الأساسي بعمل أي تغييرات في مخطط قاعدة البيانات، ولا يقدم طريقة للتحكم في أمن قاعدة البيانات "تغييرات المخطط هي تلك التي تؤثر على تصميم قاعدة البيانات، مثل إضافة أو حذف الحقول". للخروج من هذه الدائرة المحدودة، يقدم ADO امتداداً يعرف باسم the Microsoft ActiveX Data Objects. ويقدم هذا الامتداد طريقة مستندة إلى الكائن لمعالجة المخطط، حتى يستطيع المبرمج كتابة تعليمات adox البرمجية التي سوف تعمل ضد مصادر البيانات المختلفة. لمزيد من المعلومات عن استخدام ADOX انظر Access 2000 Developer's Handbook by Paul Litwin, Ken Getz, and Mike Gilbert (Sybex, 1999)

يوجد ثمانية كائنات ADO ستحتاج إلى أن تعرفها جيداً، كما هو موضح في الجدول "٦-٢٦" بالإضافة إلى هذه الكائنات، توجد أحداث تشير إلى ان هناك عمليات معينة سوف تظهر أو أنها قد تظهر بالفعل. وهذه الأحداث ممثلة في نموذج ADO كروتين معالج الحدث. انظر مقطع "ADOModel" الذي سيرد ذكره لاحقاً في هذا الفصل لتتعرف على مزيد من التفاصيل الخاصة بكائن Connection وبالأحداث وبروتين معالج الحدث.

الجدول ٦-٢٦ : كائنات نموذج ADO

الكائن	الشروح
Connection	يمثل الجو الذي سيتم فيه تبادل البيانات مع مصدر البيانات. ونظراً لأن البيانات لا تكمن في تطبيق أكسس، فإنه يجب عمل اتصال قبل البدء في اتخاذ أي إجراءات.
Command	يقدم طريقة لمعالجة مصدر البيانات يمكن إضافة أو حذف أو تحديث أو استعادة بيانات من مصدر البيانات.
Parameter	يحيط بإجراء المتغير لكائن Command. غالباً ما يتطلب الأوامر معامل ليساعد على تعريف تشغيل الأمر وهذا المعامل يكون قابل للتغيير بحيث يمكن تغييره قبل تنفيذ الأمر.
Recordset	يقوم بدور المخزن المؤقت المحلي للبيانات من مصدر البيانات.
Field	يمثل الأعمدة في جدول Recordset. تضم الحقول خصائص تساعد على تعريف الحقل، ومن أمثلة تلك الخصائص Type "String, Binary, Boolean, etc." وvalue "البيانات الحقيقية من السجل" وname.
Error	يظهر في أي وقت يواجه فيه التطبيق خطأ ما. قد تظهر الأخطاء أثناء أي مرحلة من مراحل إدارة البيانات. ولكل كائن Connection مجموعة كائنات Error الخاصة به.
Property	يعرف كائنات Connection وCommand وRecordset وField. لكل كائن ADO مجموعة خصائص تصف او تتحكم في سلوك هذا الكائن. وهناك نوعان من الخصائص هما الخصائص المضمنة والخصائص الحيوية.

الجدول ٦-٢٦: كائنات نموذج ADO

الكائن	الشرح
	تعد الخصائص المضمنة جزء من كائن ADO وتكون دائماً متوفرة. أما الخصائص الحيوية فيتم إضافتها بواسطة التزويد OLEDB وتظهر فقط وقت استخدام التزويد.
Collection	يضع مجموعات كائنات ADO المماثلة في مجموعات. وكائنات Collection الأربعة هي Errors وparameters وFields وProperties يكون للأوامر مجموعات Parameter تضم كل معامل يمكن تطبيقه على هذا الأمر. وتتكون مجموعة السجلات من مجموعة Fields تضم جميع الحقول في مجموعة السجلات تلك.

مراجع كائن ActiveX Data

تعد تسمية كائنات البيانات في نموذج ADO أكثر سهولة منها في نموذج ADO، ويكون بناء الجملة كالآتي: ADODB.objectname. فعلى سبيل المثال، للإشارة إلى كائن Recordset في ADO اكتب ADODB.Recordset. فبهذه الجملة لن تحتاج إلى عامل التشغيل نقطة التعجب. ومن ثم يمكن عمل أي مرجع باستخدام عامل التشغيل النقطة "." فقط وذلك للفصل بين المجموعات وكائنات العنصر وبناء الجملة القوسي للإشارة إلى اسم الكائن. فعلى سبيل المثال، حقل CompanyName لكائن Recordset في ADO تكون الإشارة كالآتي: ADODB.Recordset.Fields ("CompanyName").

تلميح عند إنشاء مشروع خالي، يقوم أكسس بتحميل كائن ADO فقط. وإذا أردت استخدام كائنات DAO، فستحتاج إلى إضافة مكتبة كائن DAO، في مربع الحوار References في Visual Basic Editor. لفتح Visual Basic Editor اضغط Alt+F11 لفتح مربع حوار References حدد Tools ⇦ References، ثم حدد References. Select Microsoft DAO 3.6 Object Library.

لممارسة تسمية كائنات ActiveX، سنستخدم مشروعاً خالياً يحتوي على مكتبة كائن ADO.

١-قم بتشغيل أكسس ٢٠٠٠ ثم بإنشاء قاعدة بيانات جديدة باستخدام Wizards قاعدة بيانات اكسس وصفحاتها ومشروعاتها. انقر OK.

٢-في مربع حوار New، انقر علامة الجدولة General ثم انقر "New Project DataBase" انقر OK.

٣-أعطي الملف اسم ADO Project واحفظه في مجلد C:\VBAHandBook ثم انقر Click، وسيبدأ تشغيل Microsoft SQL Server Database Wizard.

٤-اكتب "Local" في حقل "What SQL Server would you like to use for this database" ثم انقر Next. وسيفرض تحديد قاعدة البيانات المحلية على أكسس استخدام MSDE. على جهازك المحلي لا على ملقم SQL حقيقي والذي يعد استخداماً طبيعياً للمشروع. انقر Finish بعد ذلك.

ملاحظة لكي تعمل هذه الأمثلة، يجب تثبيت MSDE محلياً. فإذا لم تكن قد قمت بذلك بالفعل قم بإدراج قرص التثبيت المضغوط لأكسس ٢٠٠٠ أو بالانتقال إلى المجلد المحلي الذي تشترك فيه أيضاً ملفات التثبيت. ثم قم بتشغيل ملف SETUPSQL.EXE في مجلد \SQL\X86\SETUP وتأكد من وضع إعادة التمهيد للحاسب الآلي بعد الانتهاء من التثبيت.

٥-الآن يمكن اختبار بعض مصطلحات التسمية للكائنات في نموذج ADO اضغط Alt+F11 لفتح Visual Basic Editor وانقر زر Insert Module لفتح نافذة تعليمات برمجية.

٦-اكتب Function Test1 ثم اضغط Enter، وسيتم فتح نافذة Module أخرى لها العنوان Function Test1( ) والخاتمة End Function قم بإدخال التعليمات البرمجية للوحدة النمطية الجديدة بين هذه الأسطر.

ملاحظة يجب تعريف متغيرات الكائن في نموذج ADO أولاً قبل معالجتها. لا يمكن استخدام نافذة Immediate لتعريف المتغيرات مما يعني أن العديد من العمليات التي تؤدى في نافذة Immediate يجب أن تؤدى في نافذة Module. انظر الفصل السابع لمزيد من التفاصيل عن العمل في نافذة Module.

٧-اكتب Dim rst as New ADODB.Recordset ثم اضغط Enter. Dim هو أمر VBA الذي يعني "set aside storage for". يقوم هذا الأمر بتعريف متغير rst على أنه كائن Recordset من نوع ADO فارغ وجديد "سنقوم بتعيين قيمة Recordset إليه في الخطوة رقم ١١".

٨-نظراً لأن لدينا مشروع فارغ، فيجب علينا إما إنشاء جدول جديد وإدراج البيانات فيه أو استيراد البيانات من مصدر البيانات أخر قائم. وسنقوم بتنفيذ الإجراء الثاني. افتح نافذة Database وانقر بزر الماوس الأيمن على زر Tables ثم انقر Import.

٩-سيتم فتح مربع حوار Import. ولعل أحد ميزات ADO هو أنها يمكنها استخدام بيانات من عدة مصادر بما فيها قواعد بيانات أكسس. حدد ملف the Northwind_Ch5,6.mdb من مجلد c:\VBAHandbook directory ثم انقر Import.

١٠-إلى جانب الجدول يمكن أيضاً استيراد نماذج وتقارير وصفحات وماكرو ووحدات نمطية. حدد جدول Employees من مربع القائمة Tables ثم انقر OK، وستعود مرة أخرى إلى نافذة Database لاحظ أن جدول Employees قد اصبح الآن مدرجاً في نافذة Object.

١١-عد مرة أخرى إلى Visual Basic Editor. اكتب rst.Open "Employees", CurrentProject.Connection تحت السطر الذي قمت بكتابته في الخطوة رقم٣ ثم اضغط Enter، وسيتم تعيين قيم حقل جدول Employees في متغير rst.

١٢-في السطر التالي اكتب Debug.Print rst.Fields ("EmployeeID") ثم اضغط Enter.

١٣-اكتب rst.Close لإغلاق متغير rst، ثم اضغط Enter.

١٤-الآن اصبح لديك وحدة VBA نمطية وظيفية كاملة، فقط قم بترجمتها ثم بتشغيلها. نظراً لأنك قد قمت بإدراج سطر Debug Print فستستطيع رؤية إخراج الوحدة النمطية في نافذة Emmediate. حدد Debug ⇦ Compile ADOProject. وأثناء عملية الترجمة، سيقوم Visual Basic Editor بتنبيهك عند ظهور أي مشكلة خاصة ببناء جملة التعليمات البرمجية، فيلقي الضوء على السطر الذي به مشكلة. الآن يمكنك اختبار الوحدة النمطية الجديدة إذا افترضنا عدم وجود أية أخطاء.

١٥-اكتب Test1 في نافذة Immediate ثم اضغط Enter. ونظراً لأنك لم تنقل مؤشر السجل، فإن السجل الأول يظل هو نفسه السجل الحالي وسيتم إعادة القيمة "١".

ملاحظـة من الضروري معرفة الفرق بين إجراء الوظيفة والإجراء الثانوي عند كتابة التعليمات البرمجية. فبالنسبة لإجراء الوظيفة فيمكنه إعادة قيمـــة، أي أنـــه يمكنك تعيين نتائج الإجراء في متغير له نفس اسم الإجراء، ومن ثم يمكنـــك أداء مهام مختلفة باستخدام متغير الوظيفة. أما الإجراء الثانوي فهو يســـمح بتنفيذ الأوامر، لكنه لا يسمح بإعادة متغير. سيتم شـــرح إجـــراء الوظيفـــة والإجراء الثانوي بتفصيل أكثر في الفصل السابع.

معالجة كائن Recordset من نوع ADO

يعد العمل مع كائنات Rrecordset من نوع ADO إلـــى حـــد مـــا ممـــاثلاً لمعالجـــة كائنـــات Recordset من نوع DAO، فنموذج ADO مثل نموذج DAO في أنه يمكن استعادة البيانات من سجل واحد فقط في المرة الواحدة. لذا يكون من الضروري معرفة تغيير تركيز السجل الحـــالي من سجل إلى أخر.

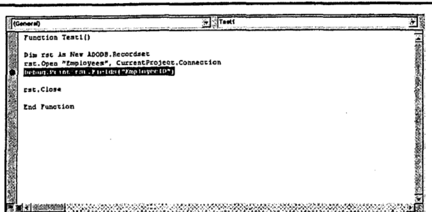
استخدام التنقل الحقيقي مع كائنات ADO

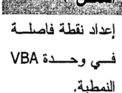
يستخدم ADO نفس طريقتي التنقل الحقيقي التي يستخدمها نموذج DAO، وهما طـــرق ....Move والإشارات المرجعية. ونظراً لأن بناء الجملة يكون مختلفاً بعض الشيء، فسوف نقوم باستكشـــاف عدة أمثلة.

استخدام طرق Move و...Move مع نموذج ADO

لشرح كيفية عمل طرق Move و...Move مع ADO، ستستخدم الوحدة النمطية "Test1" التـــي كنا قد أنشأناها من قبل. ففي أي مرة ينتهي تشغيل الوحدة النمطية، يتم إغلاق كـــائن Recordset الخاص بها. لذا سنحتاج إلى إدراج نقطة فاصلة في التعليمـــات البرمجيـــة حتـــى يظـــل كـــائن Recordset rst مفتوحاً. وعند وضع نقطة إدراج في تعليمات VBA البرمجية، فإنها تقوم بتنفيـــذ كل سطر تالي لكنها لا تقوم بتنفيذ السطر الذي يحتوي على النقطة الفاصلة.

١- انقر في الشريط الرمادي اللون الموجود بجانب الســـطر الـــذي يقـــول Debug.Print rst.Fields("Employees") ستظهر دائرة حمراء اللون بجوار السطر، وسيتم تميـــيز السطر نفسه باللون الأحمر. "انظر الشكل ٦-٥".

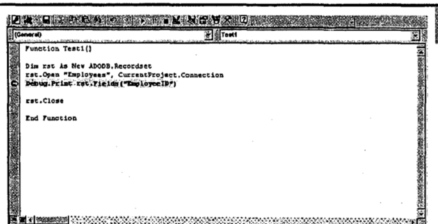




الشكل ٦-٥

إعداد نقطة فاصلة في وحدة VBA النمطية.

٢- اكتب Test1 to launch the module في نافذة Immediate لبدء تشغيل الوحدة النمطية مرة أخرى، والتي ستقف عند النقطة الفاصلة وسيتم تمييزها باللون الأصفر. "انظر الشكل ٦-٦".



الشكل ٦-٦

عندما تصل الوحدة النمطية إلى النقطة الفاصلة يقوم VBA بتمييزها "باللون الأصفر".

٣- اكتب rst.MoveNext ثم اضغط Enter، وستنتقل إلى السجل التالي في Recordset. لاختبار ذلك اكتب rst.Fields("EmployeeID") سيتم إعادة القيمة "٢".

٤- اكتب rst.Move 4 واضغط Enter، ثم اكتب rst.Fields("EmployeeID"). سيتم إعادة القيمة "٦".

استخدام خصائص BOF وEOF مع نموذج ADO

في ADO، تسلك الخصائص BOF وEOF نفس المسلك الذي تسلكانه في نموذج ADO. لنرى مثالاً على ذلك:

١- اكتب rst.MoveFirst في نافذة Immediate ثم اضغط Enter. اجر اختباراً بكتابة ?rst.Fields("EmployeeID") لتتأكد انك في السجل الأول. ومن ثم يجب إعادة القيمة "١".

٢- اكتب rst.MovePrevious ثم اضغط Enter. وكما عرفت سابقاً سينتقل بذلك مؤشر السجل الظاهري إلى موضع BOF "قبل السجل الأول".

٣- اكتب ?rst.BOF ثم اضغط Enter. وسيتم إعادة القيمة True.

إعداد الإشارات المرجعية مع ADO

لكائن Recordset من نوع ADO خاصية Bookmark التي يمكن استخدامها بنفس الطريقة التي تستخدم بها خاصية Bookmark لنموذج DAO. لترَى ذلك انظر المثال التالي:

١- اكتب rst.MoveFirst لإعادة مؤشر السجل إلى السجل الأول. يمكن اختبار ذلك بواسطة أمر ?rst.Fields("EmployeeID").

٢- اكتب rst.MoveX "حيث X هو أي رقم تختاره". وبما أن جدول Employee الذي نحن بصدده في هذا المثال، يحتوي على تسعة سجلات فقط، فينبغي عليك اختيار رقم يكون أقل من ٩، أو فيما عدا ذلك ستنتقل إلى موضع EOF. اضغط Enter.

٣- اكتب str.Mark=rst.Bookmark لتعيين متغير rst.Mark في قيمة الإشارة المرجعية في السجل الحالي. اضغط Enter. اكتب بعد ذلك ?rst.Fields("EmployeeID") لتحصل على قيمة حقل EmployeeID لسجل الإشارة المرجعية.

٤- اكتب rst.MoveX "حيث X هو أي رقم تختاره" ثم اضغط Enter.

٥- اكتب rst.Bookmark=rst.Mark ثم اضغط Enter، وسينتقل مؤشر السجل مرة أخرى إلى النقطة التي قمت بإعداد قيمة لإشارتها المرجعية في الخطوة رقم "٣".

٦- اختبر ذلك بكتابة ?rst.Fields("EmployeeID") يجب أن تحصل على نفس القيمة التي حصلت عليها في الخطوة رقم "٣".

تلميح يمكن اختبار هذا الإجراء مع أي حقل في جدول Employees باستبدال اسم الحقل بـ "EmployeeID" في هذه التعليمات.

استخدام التنقل المنطقي مع كائنات ADO

يمنح ADO طرق Seek و...Find لتحديد موضع السجلات في مجموعة من السجلات. إذا لـم تكن تعرف جيداً متى تستخدم هذه الطرق، راجع الجزء الخاص "باستخدام التنقل المنطقي" الـذي ورد ذكره سابقاً في هذا الفصل.

استخدام طريقة Seek مع ADO

يكون بناء جملة طريقة Seek في نموذج ADO كالأتي:

```
recordsetvariable.Seek(KeyValues, SeekOption)
```

حيث:

- Recordsetvariable هو اسم متغير كائن Recordset الذي قمت بإنشائه. وفي المثال الذي نحن بصدده، يشار إليه بـ rst. يجب أن تحتوي مجموعة السجلات على مجموعة فهرس واحدة على الأقل.
- KeyValues هي مصفوفة من المتغيرات تمثل قيمة ما تبحث عنه.
- SeekOption يحدد نوع المقارنة التي ترغب في عملها كالأتي:

AdSeekAfterEQ	يبحث عن مفتاح يساوي KeyValues، أو يأتي بعد المكان الـذي تمت فيه هذه المطابقة.
AdSeekAfter	يبحث عن مفتاح بعد المكان الذي قد تكون قد تمت فيه المطابقة مع KeyValues.
AdSeekBeforEQ	يبحث عن مفتاح يساوي KeyValues أو يأتي قبل المكان الذي قد تكون قد تمت فيه المطابقة.
AdSeekBefor	يبحث عن مفتاح قبل المكان الذي قد تكون قد تمت فيه المطابقة مع KeyValues.
AdSeekFirstEQ	يبحث عن المفتاح الأول الذي يساوي KeyValues.
AdSeekLastEQ	يبحث عن المفتاح الأخير الذي يساوي KeyValues.

تستخدم طريقة Seek مع خاصية Index. يجب أن يدعم التزويد الأساسي الفـهارس علـى كائن Recordset، وإلا سيكون من الحتمي استخدام طرق Find.

استخدام طرق Find مع ADO

تكمن أهمية طرق Find في تحديد مواضع السجلات التي ترضي معيار معيناً بمجموعة السجلات. ويكون بناء الجملة كالأتي:

```
Find (criteria, SkipRows, SearchDirection, start)
```

حيث:

- **Criteria**: هي قيمة سلسلة "يجب إدخالها في علامات تنصيص" تحدد اسم العمود "الحقل"، وعامل تشغيل المقارنة (<>,=<,=>,=,<,>) والقيمة التي تستخدم في البحث، وهي المعامل الوحيد الذي يكون مطلوباً في طريقة Find.
- **SkipRows**: يحدد عدد الصفوف من الصف الحالي أو من الإشارة المرجعية البادئة لبدء البحث.
- **SearchDirection**: يحدد ما إذا يكون الانتقال أثناء البحث تجاه نهاية مجموعة السجلات "adSearchForward" أم بدايتها "adSearchBackward".
- **Start**: هي إشارة مرجعية يمكن استخدامها كموضع بداية للبحث الذي تقوم به مجموعة السجلات.

فعلى سبيل المثال، للبحث عن كائن rst.Rcordset قمنا بإعداد الأتي للنموذج الأول "Janet" كقيمة لحقل FirstName في جدول Employees:

١ـ افتح نافذة Module جديدة، ثم قم بإنشاء وظيفة ثانوية تحمل اسم "Janet".

٢ـ اكتب الأسطر التالية بنفس ترتيبها في نافذة Module بعد عنوان Function:

```
Dim rst As New ADODB.Recordset
rst.Open "Employees", CurrentProject.Connection
rst.Find "FirstName = 'Janet'"
Debug.Print rst.Fields("EmployeeID")
rst.Close
```

٣ـ سيتم إعادة القيمة "٣"، موضحة أن السجل الذي يحمل فيه EmployeeID القيمة "٣" هو أول سجلات المجموعة من نقطة البداية، ويحتوي على "Janet" في حقل FirstName في كل مرة تستخدم فيها أمر rst.Open. يكون مؤشر السجل في أول حقل بحيث يبدأ البحث من هذه النقطة، إلا إذا أوضحت غير ذلك.

ملاحظة قد يكون جزء القيمة الخاص بمعيار Find سلسلة، أو رقم فاصلة عائمة أو تاريخ. فإذا كانت القيمة سلسلة فإنها تكون محاطة بعلامات تنصيص فردية، كما جاء في المثال. أما إذا كانت القيمة تاريخ، فإنه يسبقها ويأتي بعدها علامة الرقم (#) كما في #٩٩/٣/٦#.

إضافة وتحرير وحذف سجلات مع ADO

عند فتح مجموعة سجلات باستخدام وظيفة Recordset.Open يمكنك إعداد خصائص متعددة تقوم بتعريف مجموعة السجلات. فإذا أردت عمل أي تغييرات بالبيانات، فإنك ستحتاج إلى تغيير الإعدادات الافتراضية. غير أن أهم هذه الخصائص هما خاصيتي LockType وCursorType. وقد يكون لخاصية LockType أحد القيم التالية:

AdLockReadOnly	تقوم بإعداد كائن Recordset بحيث يكون مقروء فقط، ولا يمكن عمل أي تحريرات "الافتراضية".
AdLockPessimistic	تقوم بإعداد تأمين السجل بالسجل، فيتم تأمين السجلات فور التحرير.
AdLockOptimistic	تقوم بإعداد تأمين السجل بالسجل، فيتم تأمين السجلات عند استدعاء طريقة Update.
AdLockBatchOptimistic	تسمح بالتحديث الدفعي "مطلوب من أجل وضع التحديث الدفعي"

بالافتراض، تفتح مجموعة السجلات على أنها adLookRecordonly مما يعني انك لن تستطيع عمل أي تغييرات بهذه المجموعة. للتغلب على ذلك قم بتغيير هذا المعامل إلى adLockOptimistic. ويكون الاختيار بين الاثنين أو أكثر وضوحاً عدد تعدد المستخدمين، ويكون الفرق بينهما هو نفسه الفرق بين التأمين أثناء التحرير والتأمين أثناء التحديث.

أما بالنسبة لخاصية CursorType فهي توضح نوع المؤشر المستخدم في مجموعة السجلات وهو يكون أقرب إلى أنواع مجموعات السجلات DAO المختلفة. وقد يكون لخاصية CursorType أحد الإعدادات التالية:

AdOpenForwardOnly	يقوم بإعداد نسخة ثابتة من السجلات التي تستخدم لتحديد مواضع السجلات، غير أنها لا يمكن تمريرها إلا إلى الأمام "الافتراض".
AdOpenKeyset	يسمح بعمل إعدادات في مجموعة السجلات، لكن لا يستطيع المستخدمون سوى رؤية ما قد حرروه.
AdOpenDynamic	يسمح بجعل التحريرات والإضافات والتغييرات والحذوف التي يقوم بها الآخرون مرئية.
AdOpenStatic	يقوم بإعداد نسخة ثابتة لمجموعة السجلات. لا يمكن عمل أي تحريرات.

تحرير وتحديث السجلات مع ADO

لتعديل الحقول بالسجل، يجب مراعاة الشروط التالية:

- يجب إعداد خصائص LockType وCursorType إعداد مناسباً.
- يجب أن يكون الحقل قابل للتعديل.
- يجب أن يكون السجل الذي تريد تحديثه هو السجل الحالي.

وبمجرد تنفيذ هذه الشروط، سيسهل تحرير الحقول وتحديثها. وكمثال على ذلك، سنضيف قيمة جديدة للحقل في جدول Employees.

١- افتح نافذة Module جديدة، ثم قم بإنشاء إجراء ثانوي وأطلق عليه اسم NewValue.

٢- اكتب الأسطر التالية في نافذة Module.

```
Dim rst As New ADODB.Recordset
rst.Open "Employees", CurrentProject.Connection, adOpenStatic, _
adLockOptimistic
rst.Find "FirstName = 'Janet'"
rst.Fields("TitleofCourtesy") = "Mrs."
rst.Update
rst.Close
```

٣- ترجم الوحدة النمطية ثم قم بتشغيلها. بعد تحديث السجلات في جدول Employees "اختر Records ⇦ Refresh" سترى ان التغيير الذي قمت بعمله كان ناجحاً.


ملاحظــة يجب أن تعرف انه عند العمل مع كائن Recordset فإنك تعمل مع نسـخة مؤقتة من البيانات، لذا يجب تطبيق أي تغييرات تقوم بعملها مـرة أخـرى على البيانات الأصلية باستخدام طريقة Update.

إضافة وتحديث السجلات مع DAO

لإضافة سجل استخدم وظيفة AddNew لكائن Recordset لتطبيق ذلك افتـح نـافذة Module جديدة وأطلق عليها اسم AddNewRecord واكتب الأتي:

```
Dim rst As New ADODB.Recordset
rst.Open "Employees", CurrentProject.Connection, adOpenStatic, _
adLockOptimistic
rst.AddNew
rst.Fields("LastName") = "Johnson"
rst.Fields("FirstName") = "Robert"
rst.Update
rst.Close
```

ترجم الوحدة النمطية وقم بتشغيلها، ثم قم بتحديث جدول Employees وسترى السجل الجديد الذي قمت بإنشائه. يمكن إدخال بعض التعديلات البسيطة على هذا السجل لتزداد أهميته في إنشاء سجلات عديدة. ستحتاج أولاً إلى تحديد عنوان الوحدة النمطية بحيث يكون كالأتي:

```
Function AddNewRecord(strFirstName As String, strLastName As String)
```

يعرف هذا العنوان متغيران جديدان يمكـن اسـتخدامهما فـي سـجلات الإدخـال وهمـا: str.FirstName وstr.LastName والآن سنحتاج إلى تعديل الوحدة النمطية تعديلاً طفيفاً. فقـط قم بتغيير أول سطرين اللذين يبدأن بـ rst.Fields ليكونا كالأتي:

```
rst.Fields("LastName") = strLastName
rst.Fields("FirstName") = strFirstName
```

إعـداد ترجمـة الوحـدة النمطيـة، ثـم افتـح نـافذة Immediate واكتـب "AddNewRecord"Amy","Brumley. نظراً لأنك قد عرفت متغـير str.FirstName فـي البداية، ستصبح Amy هي قيمة حقل FirstName للسجل الجديد. بينما ستصبح Brumley هـي قيمة حقل LastName. يمكنك رؤية ذلك بعد تحديث السجلات في جدول Employees.

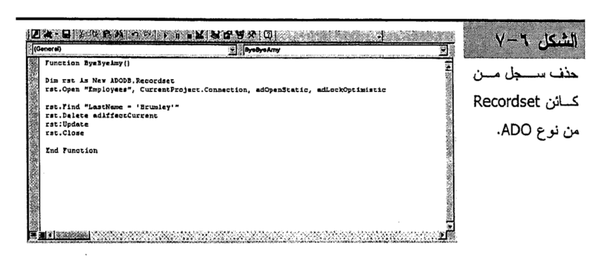
حذف السجلات مع ADO

لحذف السجل يجب أن تجعله أولاً سجلاً حالياً، ثم استخدم وظيفـة Delete لكـائن Recordset لحذف السجل من مجموعة السجلات المخزنة. وكخطوة أخيرة استخدم وظيفة Update لمجموعة السجلات لتوصيل التغييرات إلى مصدر البيانات.

وكمثال على ذلك، سنقوم بحذف سجل Amy Brumley الذي أنشأناه. افتح نافذة Module جديدة تعرف باسم ByeByeAmy ثم اكتب الأسطر التالية:

```
Dim rst As New ADODB.Recordset
rst.Open "Employees", CurrentProject.Connection, adOpenStatic, _
adLockOptimistic
rst.Find "LastName = 'Brumley'"
rst.Delete adAffectCurrent
rst.Update
rst.Close
```

يجب أن تكون الوحدة النمطية كما هي موضحة في الشكل "٦-٧".



الشكل ٦-٧

حذف سجل من كائن Recordset من نوع ADO.

ترجم الوحدة النمطية وقم بتشغيلها، ثم قم بتحديث السجلات في جدول Employees. سترى ان سجل Amy Brumley لم يعد جزءاً من مجموعة السجلات.

نموذج ADO

لقد كانت لك بالفعل بعض التعاملات مع كائنات ActiveX Data كما رأيت توضيحاً لتسلسل ADO الهرمي. يوضح الجدول "٦-٣٦" الكائنات الثمانية لنموذج ADO. لقد عملنا في هذا الفصل مع كائن Recordset من نوع ADO واستخدمنا بعض خصائصه وطرقه في الأمثلة. غير أن خصائص وطرق كائن Recordset تحتاج إلى بعض الوقت لتقرأ عنهم في ملفات تعليمات مايكروسوفت أكسس. "وعن كائنات ADO الأخرى التي سوف تستخدمها". في الفصول التالية سيكون التركيز على العمل مع كائن Recordset من نوع ADO، فستقدم تفاصيل أكثر عن كائن Connection وعن الأحداث وعن روتين معالج الحدث.

كائن Connection

يوفر كائن Connection الجو المطلوب توافره لتبادل البيانات بين مصدر البيانات وتطبيق أكسس. والإجراء هو سلسلة من عمليات تشغيل البيانات التي تظهر عبر الاتصال. والعمليات في الإجراء جميعها "جميعاً أو لا شيء" مما يعني أنه في حالة إخفاق أي عملية من هذه العمليات لن يتم عمل أي تغييرات بمصدر البيانات. لذا، إذا قمت بإلغاء إجراء بعد بلوغ أي مرحلة، فيكون الأمر وكأنه لم يتم إجراء أي عمليات. وتتضمن طرق كائن Connection عمليات الإجراء. يوضح الجدول "٦-٣٧" طرق الكائن Connection.

الجدول ٦-٢٧: طرق كائن Connection

الطريقة	الشرح
BeginTrans	تبدأ إجراء جديد.
CommitTrans	تحفظ التغييرات وتنهي الإجراء الحالي. ويمكنها أيضاً بدء إجراء جديد.
RollbackTrans	تلغي التغييرات التي تم إدخالها أثناء الإجراء الحالي، تنهي الإجراء.
Execute	تنفذ الاستعلام المحدد أو عبارة SQL أو الإجراء المخزن أو تزويد النص المحدد.
Open	تفتح اتصالاً بمصدر البيانات.
OpenSchema	تستخلص معلومات عن مخطط قاعدة البيانات من التزويد.

يستطيع نموذج ADO الاتصال بالبيانات أو تشغيلها من أي تزويد OLE DB. ويستخدم كائن Connection لتحديد تزويد OLE DB ولتحديد أي معامل ضروري للاتصال بالتزويد، مثل الرقم المعروف للمستخدم وكلمة المرور.

الأحداث وروتين معالج الحدث

تشير الأحداث إلى أن هناك عمليات معينة على وشك الحدوث أو حدثت بالفعل. وتجسد هذه الأحداث في نموذج ADO روتين معالج الحدث، واستدعاء معالج الحدث قبل حدوث العمليات

يعطي للمبرمج الفرصة لفحص وتعديل معامل العملية أو لإلغاء العملية أو للسماح بمواصلتها. أما استدعاء معالج الحدث بعد انتهاء العمليات، فهو يعطي ملحوظة إلى استكمال عمليــــة لاتزامنيـــة. وهناك عائلتان للأحداث:

Connection Events: وهي الأحداث التي يتم إصدارها عند بدء الإجراءات أو حدوثــها أو تراجعها، وعند قيام الأوامر بالتنفيذ وعند بدء الاتصالات أو انتهائها.

Records Events: وهي الأحداث التي يتم إصدارها لتقرير مدى تقـــدم عمليـــة اســتعادة البيانات، وتقرير متى يتم التنقل في مجموعة السجلات ومتى يتم تحرير مجموعة السجلات.

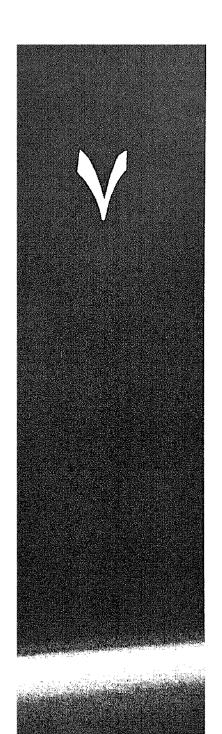
خلاصة

لقد تعرفت في هذا الفصل على كائنات تشغيل البيانات التي تمثل عنـــاصر محــرك قـــاعدة Jet للبيانات التي يمكن معالجتها في إجراءات VBA. كما تعرفت أيضاً علـــى كائنـــات ADO التـــي تديرها MSDE عند العمل في مشروع أكسس. والنقاط التالية هي أهم النقاط:

- ♦ عند الإشارة إلى كائن تشغيل بيانات في إجراء VBA، ستبدأ من قمة التسلســل الــهرمي لكائن تشغيل البيانات ثم تعبر إلى الكائن في نموذج DAO.
- ♦ عند الإشارة إلى كائن ADO في إجراء VBA، قم أولاً بإعداد متغير لكـــائن Recordset ثم تسمية باقي الكائنات الأخرى بعد هذا المتغير مستخدماً الفاصل النقطة ".".
- ♦ لكائن البيانات خصائص مضمنة أنشأتها قاعدة البيانات وقد يكون له أيضاً خصائص قــام بإنشائها التطبيق المضيف وخصائص أخرى تقوم بإنشائها بنفسك. ويمثل كـــل خاصيـــة كائن Property الذي تحتوي عليه مجموعة Properties للكائن.
- ♦ يمكن إنشاء كائنات تشغيل بيانات جديدة في نموذج ADO. ويكون لكل كائن إنشاء محـــدد وقواعد تدمير: فبعض الكائنات تتطلب إنشاء الكائنات الفرعية قبـــل حفــظ التغيـــيرات، وبعضها يجب أن يضاف إلى المجموعات الخاصة بهم قبل حفظ التغييرات وبعضها قـــد يتم إنشائه من جديد ويدمر تلقائياً عند انتهاء الإجراء أو عند إغلاق قاعدة البيانات.
- ♦ استخدم كائن Recordset للعمل مع البيانات في قاعدة البيانات وهنـــاك أربعـــة أنـــواع لكائنات Recordset مـــن نـــوع DAO، وهـــي: المجموعـــة الحيويـــة، والجـــدول، وsnapshot، وإلى الأمام فقط.
- ♦ عند إنشاء مجموعة سجلات جديدة، يمكن العمل مع سجل واحد فقط "يعرف باسم السجل الحالي" في المرة الواحدة.

- يمكن التنقل عبر سجلات المجموعة مستخدماً طرق ...Move. يمكن استخدام خصائص EOF وBOF لتحديد ما إذا كنت قد تخطيت حدود مجموعة السجلات.
- يمكن تحديد موضع سجل محدد، مرضياً بذلك معيار البحث مستخدماً طريقة Seek لمجموعة سجلات الجدول، وطريق ....Find لمجموعة سجلات المجموعة الحيوية وsnapshot كما يمكن استخدام خاصية NoMatch لتحديد ما إذا كانت عملية البحث ناجحة أم لا.
- يمكن استخدام كائنات تشغيل البيانات في إجراءات VBA لإضافة سجلات جديدة أو تعديل السجلات الموجودة بالفعل أو حذفها.
- يمكن استخدام طريقة Clone لتكرار كائن Recordset من نوع DAO. ويكون لكائن Recordset المكرر نفس الإشارات المرجعية للكائن الأصلي.

لقد قدم الفصلان الخامس والسادس العديد من نماذج الكائنات التي تحتوي على الكائنات التي تريد معالجتها في برنامج VBA. وبمجرد أن تتعلم مما سيلي هذين الفصلين كيفية كتابة برامج، سيكون هذان الفصلان مرجعاً تلجأ إليه لترى ما إذا كان الكائن يحتوي على خاصية أو طريقة تحتاجها في برنامجك لكن قبل استخدام الخاصية أو الطريقة أول مرة، ابحث عن العنصر في عارض الكائن "Object Browser" أو في التعليمات الفورية لتتعرف على مزاياه ونقائصه.



كتابة الإجراءات

هذا هو الفصل الأول من ثلاثة فصول تشرح كل المميزات الأساسية الخاصة بلغة برمجة VBA كما تستخدم في أكسس ستعرف في هذا الفصل أساسيات كتابة الإجراءات. يبدأ الفصل بمراجعة العبارات التي تعلمتها حتى الآن ويلي ذلك مناقشة لأنواع بيانات VBA الموجودة في أكسس ثم يقدم الفصل آليات كتابة الإجراءات مثل أنواع الإجراءات التي يمكن كتابتها وما إذا كان التخزين سيتم في ملفات قاعدة المعلومات وكيفية تشغيلها في الفصل الثامن، ستتعرف كيفية استخدام المتغيرات والثوابت في الإجراءات في الفصل التاسع، ستتعرف على التحكم في التنفيذ وعلى الكثير من العبارات المفيدة والوظائف الموجودة في أكسس.

إجراءات وعبارات VBA

العبارة هي المربع من العبارات الأساسية والثوابت والمتغيرات ورموز عامل التشغيل والكائنات والخصائص والوسائل التي تعبر عن تعريف واحداً أو إعلان أو عملية واحدة يجب أن يتم مزج هذه العناصر بصورة صحيحة تبعاً لقواعد بناء جملة مترجم VBA قبل تنفيذ العبارة في إطار Immediate، تستخدم سطر واحد لكل عبارة وفيما يلي بيان منفصل عن العبارات التي تعلمتها في الفصلين ٥ و٦.

الوصف	بناء الجملة
طبع قيمة ما	? value Debug.Print value or Print value
تعيين قيمة لخاصية	object.property = value
تعيين قيمة خاصية لمتغير	Let var = object.property or var = object.property
تعيين كائن تم إعادته بواسطة خاصية لمتغير كائن	Set obj = object.property
استدعاء طريقة دون وجود أي شيء معاد	object.method argumentlist
استدعاء طريقة بوجود قيمة معادة	Let var = object.method (argumentlist) or var = object.method (argumentlist)
استدعاء طريقة بوجود كائن معاد	Set obj = object.method (argumentlist)

إطار Immediate أداة رائعة لتشغيل عبارات VBA الفردية وتستخدم برمجة VBA حزم عبارة أو أكثر داخل وحدة تدعى إجراء أو سؤال الحاسب الآلي بتشغيل الإجراء.

أنواع البيانات

تتم الإشارة لأنواع القيم التي تحملها المتغيرات كأنواع بيانات عندما تعمل تفاعلياً مع أكسس، تقوم بتحديد أنواع بيانات عندما تقوم بإنشاء حقول جدول لكن بخلاف ذلك لا تحتاج للاهتمام بها كثيراً نوع البيانات مفهوم مهم في برمجة VBA. عندما تقوم بإنشاء متغير ليحمل قيمته أو ليشير إلى كائن وتستطيع ترك VBA ليعالج نوع البيانات هذا لأجلك. VBA له نوع بيانات صالح لكل الأغراض يدعى نوع بيانات متغير Variant والذي يقوم VBA باستخدامه إلا إذا حددت واحداً أخر. مشاكل هذا الاتجاه هي أي المتغيرات التي بها نوع بيانات Variant تأخذ ذاكرة كبيرة وإجراءات VBA التي تستخدمها تعمل ببطء. لذلك، لإنشاء برامج تستخدم ذاكرة وتعمل بأسرع ما يمكن، ستحتاج لتعلم بعض الأشياء عن أنواع البيانات.

ملاحظة موضوع أنواع البيانات معقد نتيجة الحقيقة التي تتمثل في وجود خمسة أنظمة لأنواع البيانات التي ستتعامل بها وهي أنواع حقل البيانات التي تحددها عندما تقوم بإنشاء الجداول تفاعلياً، وأنواع بيانات VBA وDAO وADO التي ستتعرف عليها في هذا الفصل وأنواع البيانات.

أنواع البيانات الأساسية

أنت معتاد على أنواع بيانات الحقل الذي تعنيه لحقول الجدول عندما تقوم بإنشاء جداول أكسس. VBA أكسس له مجموعته الخاصة من أنواع بيانات VBA والتي تلائم معظمها أنواع بيانات الحقل يسرد الجدول ٧-١ أنواع بيانات VBA وعلامات الاسم شائعة الاستخدام وانواع بيانات الحقل المتناسقة ووصف مختصر لكل نوع بيانات الحقل المتناسقة ووصف مختصر لكل نوع بيانات هناك نوعا بيانات VBA ليس لهما نظير في أنواع بيانات الحقل وهما Variant وObject وستتعرف على أنواع البيانات هذه في الأقسام التالية:

الجدول ٧-١: أنواع البيانات الأساسية في VBA .

نوع بيانات VBA	علامة الاسم	نوع بيانات الحقل المقابل	حجم التخزين	الوصف
Byte	byt	Number (Byte)	1 byte	ثنائي صفر إلى ٢٥٥ "لا يوجد كود"
Integer	int	Number (Integer)	2 bytes	رقم -٣٢، ٧٦٨ إلى ٣٢ ٧٦٧ "لا يوجد كود" للعمليات الحسابية أسرع مع الأرقام.
Long	lng	4 bytes	Number (Single)	رقم -٢، ١٤٧، ٤٨٣، ٦٤٨ إلى ٢، ١٤٧، ٤٨٣ن ٦٤٧ (لا يوجد كود)
Single	sng	Number (Single)	4 bytes	دقة فردية الفاصلة العائمة. عُرضة لأخطاء تقريبية صغيرة
Double	dbl	Number (Double)	8 bytes	دقة زوجية فاصلة عائمة عُرضة لأخطاء تقريبية صغيرة
Decimal		Number (Double)	12 bytes	رقم غير مرمز مدرج لأس ١٠ محدداً عدد الأرقام إلى يمين النقطة العشرية الدقة العشرية إلى ٢٨
Currency	cur	Currency	8 bytes	رقم مدرج، فاصلة عائمة يستخدم إلى أربعة أرقام عين الرقم العشري و١٥ إلى اليسار نوع بيانات محدد النقطة ودقيق

الجدول ٧-١: أنواع البيانات الأساسية في VBA

نوع بيانات VBA	علامة الاسم	نوع بيانات الحقل المقابل	حجم التخزين	الوصف
String (fixed length)	str	Text and Memo	10 bytes+string length	سلسلة صفر إلى ٢ بليون حرف فرضياً، متغير سلسلة هو سلسلة متغيرة الأطوال
String (fixed length)	str		String length	١ إلى ٦٥,٤٠٠ حروف. في وحدات النموذج أو التقرير يجب إعلان السلاسل المحددة الطول Private.
Boolean	bln	Yes/No	2 bytes	تم التخزين كأحرف ٢ بايت ولكي نقيم True أو False القيمة الفرضية هي False. عندما يتم التحويل إلى أنواع بيانات أخرى يصبح False ٠ وTrue 1
Date	dtm	Date	8 bytes	كانت مثل مستند وورد أو صورة أو ملف صوت مضمن أو مربوط في جدول
String		OLE Object	Up to 1 gigabyte	نص أو مزيج من نص وأرقام مخزنة كنص ومستخدمة كعنوان ارتباط تشعبي
String		Hyperlink		يحمل القيم المستخدمة للبحث عن قيم في جدول أو قائمة أخرى

الجدول ٧-١: أنواع البيانات الأساسية في VBA

نوع بيانات VBA	علامة الاسم	نوع بيانات الحقل المقابل	حجم التخزين	الوصف
N/A		Lookup Wizard		حتى ٢٠٤٨ حرفاً لكل أجزاء العنوان الثلاثة
Object	obj		4 bytes	نفس الحجم مثل حقل المفتاح الأولى
Variant (with n-umbers)	var		16 bytes	يحمل أنواعاً مختلفة من البيانات أية قيمة رقمية حتى نطاق Double
Variant (with ch-aracters)	var		22 bytes+stnin g length	سلسلة صفر إلى ٢ بليون حرف

متغيرات إعلان النوع

يُفضل بعض المبرمجين استخدام حرف إعلان نوع ملحق باسم متغير لتحديد نوع بيانات المتغير. أنواع البيانات التي لها مثل هذه الأحرف وأحرف إعلان النوع الخاصة بها هي ما يلي:

- Integer (%)
- Long (&)
- Single (!)
- Double (#)
- Currency (@)
- String ($)

حرفي

عندما تقوم بتعيين قيم حرفية لمتغير له نوع من أنواع البيانات الأساسية. يجب عليك اتباع القواعد التالية:

String: ضمن سلسلة حرفية في علامات تنصيص مزدوجة على سبيل المثال، strName = "Betsy Evart".

Date: ضمن تاريخ حرفي في علامات رقم على سبيل المثال، = dtmOrderDate .#5/10/96#

Currency: أحذف علامة الدولار والفواصل على سبيل المثـــال، = curAmount .3200

نوع بيانات Variant

نوع بيانات Variant هو نوع بيانات متقلب بمعنى أنه يستطيع تخزين أي نوع بيانات أساس أخر باستدعاء نوع بيانات String المحدد الأول. والغرض من نوع بيانات Variant هو السماح لـــك باستغلال قيم بأنواع بيانات مختلفة دون الحاجة لتحويل أنواع البيانات الخاصة بها بنفسك. نـــوع بيانات Variant هو نوع بيانات فرضي في أكسس في VBA لذلك إذا لم تحدد بوضـــوح نـــوع البيانات لمتغير ما، يقوم VBA بتعيين نوع بيانات. عندما تعمـــل فـــي إطـــار Immediate، لا تستطيع تحديد نوع بيانات بوضوح باستخدام عبارات إعلان نوع البيانات التـــي ســـتعرفها فـــي الفصل التالي. يمكنك تحديد نوع بيانات متغير في إطار Immediate مستخدماً أحرف إعـــلان النوع لأنواع البيانات هذه والتي تحتوي عليها (كما تم سردها في القسم السابق) إذا لـــم تســـتخدم أحرف إعلان النوع، يكون للمتغيرات التي تقوم بإنشائها في إطار Immediate نـــوع بيانـــات Variant.

المميزات الأساسية لاستخدام متغيرات Variant تتضمن ما يلي:

- من الأسهل إنشاء الشفرة لأنك لا تحتاج لمعرفة أنواع البيانات ولا للقلق مـــا إذا كـــانت أنواع البيانات الخاصة بالمتغيرات متوافقة أم لا.
- متغير بيانات Variant هو الوحيد الذي له قيمة Null إذا أردت تعيين متغـــير لحقـــول ويمكن أن يكون لها قيمة Null، فإن استخدام متغير Variant يجنـــب الحاجـــة لكتابـــة الشفرة للتعامل مع قيم بيانات Null "تمت مناقشة قيم Null لاحقاً".

أما عيوب استخدام متغيرات Variant فهي:

- تحتاج أقصى ذاكرة كلما كبرت الذاكرة المربوطة في متغيرات الحفظ، كلما قلت فرصـــة تواجدها لتشغيل التطبيق.
- تتم الإجراءات التي تستخدم متغيرات Variant ببطء في كل مرة تشفير فيـــها لمتغـــير Variant، يجب على VBA أن يأخذ وقته في تحويل نوع البيانات لمتغير كائن لتحديد ما إذا كانت الخصائص والطرق التي تم الإشارة إليها في الشفرة ملائمة للكائن أم لا.

تحويل نوع البيانات

يسعى VBA تلقائياً للتحويل بين أنواع البيانات عند الضرورة لعملية حسابية ما. بالطبع لا يكون التحويل ممكناً دائماً وهناك بعض القيود على العمليات التي تجريها مع نوع بيانات Variant. على سبيل المثال، لإجراء عمليات حسابية على متغيرات Variant، يجب أن تحتوي المتغيرات على أرقام صالحة وإلا يتم توليد خطأ وقت التشغيل. يمكن استخدام وظيفة IsNumeric المضمنة لتحديد إذا كانت قيمة متغير Variant رقماً صالحاً قبل إجراء العملية الحسابية. بالمثل، إجراء حساب بيانات على متغيرات Variant يتطلب احتواء المتغيرات على قيم وقت وتاريخ صالحة. يمكن استخدام وظيفة IsDate المضمنة لتحديد ما إذا كان متغير Variant يحتوي على قيمة يمكن تحويلها إلى بيانات قبل إجراء حساب بيانات.

تعتمد نتيجة استخدام عامل تشغيل زائد(+) مع متغيرين Variant. إذا احتوى كل من متغيري Variant على سلاسل، يقوم عامل تشغيل + بسلسلة السلاسل، إذا احتوى كل من متغيري Variant على أرقام يقوم عامل تشغيل + بجمع الأرقام إذا كانت إحدى القيم رقماً والأخرى سلسلة، يعمد VBA إلى تحويل السلسلة إلى رقم وإذا نجح التحويل يقوم VBA بجمع الرقمين وإلا يولد VBA خطأ. تستطيع تجنب الغموض باستخدام عامل تشغيل علامة الجمع & عندما تريد سلسلة، سلسلتين واترك مسافة بين أسماء المتغير وبين عامل تشغيل &.

لاكتشاف هذه المفاهيم، افتح إطار Immediate وأدخل كلاً من التعبيرات التالية:

- اطبع varone = 4 واضغط Enter يتم إنشاء متغير Variant المسمى varone ويحمل الرقم ٤.
- اطبع IsNumeric(varone) ? واضغط Enter. يتم عرض القيمة True.
- اطبع vartwo = "3" واضغط Enter. يتم إنشاء متغير Variant المسمى vartwo ويحمل السلسلة "٣".
- اطبع IsNumeric(vartwo) ? واضغط Enter. يتم عرض قيمة True.
- اطبع ?varone+vartwo واضغط Enter يتم عرض القيمة 7.
- اطبع ?varone & vartwo واضغط Enter يتم عرض القيمة 43 كنتيجة لأن VBA حول varone لسلسلة ثم قام بسلسلة الناتجين.

يوفر أكسس مجموعة من وظائف تحويل نوع البيانات التي تستطيع استخدامها لفرض أو إجبار تعبير رقمي أو سلسلة لنوع بيانات محدد. هذه الوظائف موجودة في جدول ٧-٢.

الجدول ٧-٢: وظائف تحويل نوع البيانات

الوظيفة	الوصف
Asc(expression)	ترجع شفرة حرف ANSI "رقم" للحرف الأول في سلسلة
CBool(expression)	ترجع False إذا كان التعبير صفراً وtrue إن كان غير ذلك ترجع Cbool(3<5) True
CByte(expression)	تحول تعبيرا إلى بايت CByte(12.345) ترجع ١٢
CCur(expression)	تحول تعبيرا إلى Currency CCur(12.345678) ترجع 12.3457
CDate(expression)	تحول حروف الوقت والتاريخ وبعض الأرقام إلى Date. Date ترجع Cdate(#2/6/1943#) 2/6/43.
CDbl(expression)	تحول تعبيرا إلى Double تستخدم لتتطلب حسابات دقة مزدوجة
CDec(expression)	تحول تعبيرا إلى Decimal
CInt(expression)	تحول تعبيرا إلى Integer. تستخدم لإجبار حسابات الرقم. CInt(12.2345) ترجع ١٢
CLng(expression)	تحول تعبيرا إلى Long. تستخدم لإجبار حسابات الرقم CLng(31234.65) ترجع 31235
CSng(expression)	تحول تعبيرا إلى Single.تستخدم لفرض حساب فردي الدقة. CSng(12.3422356) ترجع 12.34224
CStr(expression)	تحول تعبيرا إلى String تحدد البيانات الموجودة في التعبير ما يرجع CStr(12.3422) ترجع السلسلة "12.3422"
CVar(expression)	تحول تعبيرا إلى Variant. Cvar(1234 & "5678") ترجع السلسلة 12345678
CVErr	تحول رقم خطأ إلى Variant
Fix	تزيل الرقم الكسري من رقم وترجع الجزء الرقمي إذا كان الرقم سالباً فإنها ترجع الرقم الأول أكبر من الرقم
Int	تزيل الجزء الكسري من الرقم وترجع الجزء الرقمي إذا كان الرقم سالباً، فإنها ترجع الرقم الأول أصغر من الرقم

قيمة Null

لنوع بيانات Variant قيمتين مميزتين وهما Null وEmpty الغرض من قيمة Null هو تحديد البيانات المفقودة أو غير المعروفة أو التي غير قابلة للتطبيق. قيمة Null المتميزة ليست قيمة حقيقية مثل "Peacock" أو $12.45 لكنها مؤشر على أن البيانات مفقودة أو غير معروفة أو لا تطبق. تحتوي البيانات الموجودة في جدول أو حقل استعلام أو في تحكم تقرير أو نموذج على نوع بيانات Variant فرضياً. عندما تترك حقلاً أو تحكماً فارغاً فإن Null يتم تخزينها تلقائياً. أنت معتاد على طرق تحدد ما إذا كان حقل أو تحكم يحتويان على قيمة Null. على سبيل المثال، تستطيع استخدام للبحث عن سجلات لها قيمة Null في حقل ما عن طريق إعداد خلية المعيار للحقل إلى IsNull وتستطيع تحديد إذا كان تحكم يحتوي على قيمة Null باستخدام وظيفة IsNull.

تتطلب قيمة Null تعاملاً مخصوصاً للأسباب التالية:

- إذا قام أي جزء من التعبير بتقييم قيمة Null، يكون للتعبير بأكمله قيمة Null. وهذا يدعى نشر قيم Null بالإضافة إلى ذلك، قيمت وسيطة لوظيفة مضمنة وظيفة تخصيص إلى Null، ترجع الوظيفة قيمة Null على سبيل المثال، إذا استخدمت SQL أو وظيفة مجال إجمالي لحساب قيمة خلاصة لحقل لمجموعة من السجلات، لا يتم تضمين السجلات التي بها قيم Null في الحقل.
- عندما تصل جداولك في استعلام ما، لا يتم تضمين السجلات التي لها قيم Null في حقل الوصل في نتيجة الاستعلام. على سبيل المثال، إذا قمت بإنشاء استعلام على جداول Orders وEmployees في Northwind، لا يتم تضمين سوى السجلات الخاصة بالموظفين الذين يتعاملون مع الأوامر والسجلات الخاصة بالموظفين الذين يتعاملون مع الأوامر والسجلات الخاصة بالأوامر التي يتم التعامل معها من قبل موظفين محدودين ولا تحتوي نتيجة الاستعلام على سجلات للموظفين الذين لم يتعاملوا مع الأوامر أو على أوامر لم تعين لموظف ما.
- عندما تقوم بإنشاء علاقة وتقوم بفرض تمام مرجعي، تظل قادراً على إنشاء سجلات وحيدة في جدول التابع عن طريق ترك حقل الوصل خالياً في الجدول التابع. في قاعدة بيانات Northwind. تستطيع إضافة ترتيبات وحيدة جديدة بترك حقول EmployeeID وCustomerID خالية.

في VBA، تحتوي متغيرات Variant فقط على قيم Null. يمكنك تحديد إذا كان متغير Variant يحتوي على قيمة Null باستخدام وظيفة IsNull. يمكنك إعداد متغير Variant لقيمة Null باستخدام عبارة تعيين كما يلي:

```
varname = Null
```

لاكتشاف قيم Null، افتح قاعدة بيانات Northwind اضغط Ctrl+G لعرض إطار Immediate، وادخل كلاً من التعبيرات التالية:

- اطبع varone = 4 واضغط Enter يحمل متغير Variant العدد الصحيح ٤
- اطبع vartwo = null واضغط Enter يحمل متغير Variant قيمة Null
- اطبع varone + vartwo? واضغط Enter يتم عرض Null كنتيجة لأن قيمة Null في vartwo تم نشرها للمجموع
- اطبع DCount("Fax", "Suppliers")? واضغط Enter يتم عرض القيمة ١٤ كرقم المزودين بقيمة مدخلة كرقم فاكس

قيمة Empty

الغرض من قيمة Empty هي كونها حرفاً ثانياً عندما تكون لم تقم بتعيين قيمة لمتغير بنوع بيانات Variant. عندما تقوم بإنشاء متغير Variant يكون له قيمة Empty حتى يتم تعيين قيمة له يمكنك استخدام وظيفة IsEmpty لتحديد إذا كان لمتغير Variant قيمة Empty. بعد تحديد قيمة لمتغير Variant، يمكنك إعادة متغير Variant مرة أخرى لقيمة Empty باستخدام عبارة التعيين التالية:

```
varname = Empty
```

قيمة Empty ليست مثل صفر أو سلسلة ذات طول صفر ("") أو قيمة Null. مع ذلك، لأن VBA يحول متغيرات Variantتلقائياً لنوع البيانات المطلوب في الاستغلال. عندما تستخدم متغير Variant بقيمة Empty في التعبيرات يحول VBA تلقائياً القيمة إلى صفر إذا تطلب التعبير قيمة رقمية أو السلسلة ذات طول صفر ("") إذا تطلب التعبير قيمة سلسلة.

للحصول على خبرة في المتناول مع هذه المفاهيم، ادخل التالي في إطار Immediate:

- اطبع varthree = "test" واضغط Enter. يتم إنشاء متغير Variant وتعيينه في اختبار قيمة السلسلة.
- اطبع IsEmpty(varthree)? واضغط Enter يتم عرض قيمة False لأن المتغير يحمل قيمته.
- اطبع varthree = Empty واضغط Enter يتم عرض قيمة True يحمل المتغير الآن قيمة Empty.
- اطبع IsEmpty(varthree)? واضغط Enter لأن المتغير يحمل قيمة True.
- اطبع varthree = Null واضغط Enter يحمل المتغير الآن قيمة Null.

- اطبع IsNull(varthree)? واضغط Enter يتم عرض قيمة True لأن المتغير يحمل قيمة Null.
- اطبع IsEmpty(varthree)? واضغط Enter يتم عرض قيمة False لأن المتغير يحمل قيمة Null ولا قيمة Empty.

استخدام وظيفة VarType

حيث يستطيع متغير Variant جعل أنواع بيانات مختلفة ويستطيع نوع البيانات المتغير في كل مرة تقوم فيها بتعيين قيمة للمتغير، تحتاج لطريقة تحدد ما نوع البيانات التي يحملها المتغير حالياً يمكنك استخدام وظيفة VarType لهذا الغرض. ترجع وظيفة VarType قيمة عدد صحيح تقابل لنوع البيانات الحالي الخاص بالمتغير. يوضح جدول ٧-٣ بعض قيم العدد الصحيح وثوابت مقابلة صحيحة تقوم وظيفة VarType بإرجاعها.

الجدول ٧-٣: قيم تم إرجاعها بواسطة وظيفة VarType

Constant	Return Value	Variant Variable Contents
Empty	0	vbEmpty
Null	1	vbNull
Integer	2	vbInteger
Long integer	3	vbLong
Single-precision, floating-point number	4	vbSingle
Double-precision, floating-point number	5	vbDouble
Currency	6	vbCurrency
Date	7	vbDate
String	8	vbString
Object	9	VbObject
Boolean	11	VbBoolean

الجدول ٧-٣: قيم تم إرجاعها بواسطة وظيفة VarType

Constant	Return Value	Variant Variable Contents
Byte	17	VbByte

يمكنك اختبار وظيفة VarType في إطار Immediate عن طريق إدخال التعبيرات التالية:

- اطبع varone = Empty واضغط Enter. يحمل متغير Variant الآن قيمة Empty.
- اطبع ?vartype(varone) واضغط Enter الناتج صفر.
- اطبع "varone = "test واضغط Enter. يحمل متغير Variant الآن قيمة سلسلة.
- اطبع ?vartype(varone) واضغط Enter الناتج ٨.
- اطبع #varone = #6/15/99 واضغط Enter يحمل المتغير Variant الآن قيمة تاريخ.
- اطبع ?vartype(varone) واضغط Enter الناتج ٧.
- اطبع varone = False واضغط Enter يحمل متغير Variant الآن قيمة Boolean.
- اطبع ?vartype(varone) واضغط Enter الناتج ١١.

نوع بيانات Object

تستخدم متغير كائن للإشارة أو الرجوع إلى كائن. يمكنك استخدام نوع بيانات Object كنوع بيانات لمؤشر الكائن (المؤشر أو المرجع هو عنوان ذاكرة ٤ بايت للكائن). نوع بيانات كائن الموجود في جدول ٧-١ عام ويمكن استخدامه لكائن ما مع ذلك يوجد أنواع بيانات كائن محددة تستطيع استخدامها بدلاً من ذلك. يسرد جدول ٧-٤ أمثلة لأنواع بيانات الكائن وعلامات اسم شائع استخدامها في أنماط تسمية Hungarian وكائنات قاعدة بيانات مقابلة، يؤدي استخدام أنواع بيانات كائن محددة إلى تنفيذ أسرع لأن VBA يعلم أن الخصائص والطرق ملائمة لكائن له نوع بيانات كائن محدد ولا تأخذ وقتاً في أثناء التنفيذ لتحديد ما إذا كانت خاصية أو طريقة مستخدمة في الشفرة ملائمة أم لا.

الجدول ٧-٤: أمثلة لأنواع بيانات Object

نوع بيانات Object	علامة الاسم	كائن قاعدة المعلومات المقابل
Database	Db	قاعدة معلومات مفتوحة
TableDef	Tdf	تعريف جدول
QueryDef	Qdf	تعريف استعلام
Recordset	Rst	تقديم في ذاكرة لمجموعة من السجلات في جدول أو سجلات نتجت من تشغيل استعلام أو عبارة SQL
Field	Fld	حقل في جدول أو استعلام أو مجموعة سجل أو فهرس أو علاقة
Index	idx	فهرس لجدول
Form	frm	نموذج أو نموذج فرعي
Report	rpt	تقرير أو تقرير فرعي
Control	ctl	تحكم عام في نموذج أو تقرير
TextBox	txt	تحكم مربع نص
ComboBox	cbo	تحكم مربع سرد وتحرير
CommandButton	cmd	تحكم زر أمر
Connection	con	كائن وصل ADO
Command	cmd	كائن أمر ADO
Parameter	par	معلم لأمر

يمكنك استخدام وظيفة IsObject لتحديد ما إذا كان المتغير يشير إلى كائن ترجع وظيفة IsObject قيمة True إذا أشار المتغير إلى كائن وقيمة False إن لم يشير إلى كائن.

قيمة Nothing

نوع بيانات Object له قيمة خاصة تدعى قيمة Nothing الغرض من قيمة Nothing هو العمل كنائب حرف عندما تكون لم تقم بتعيين كائن لمتغير بنوع بيانات Object وتتشابه قيمة

Nothing مع قيمة Empty الخاصة بنوع بيانات Variant عندما تقوم بإنشاء متغير كائن لأول مرة، تكن له قيمة Nothing حتى تعين كائناً له. بعد تعيين كائن لمتغير Object، يمكنك إعادة المتغير لقيمة Nothing بحيث لا يشير بعد ذلك للكائن باستخدام عبارة تعيين

```
Set objMyObject = Nothing
```

تستطيع استكشاف هذه المفاهيم في إطار Immediate في أثناء عملك مع الكائنات في إطار Immediate، يكون للمتغير الذي تقوم بإنشائه للإشارة لكائن نوع بيانات Variant بنوع الكائن الفرعي افتح نموذج Employees في طريقة عرض Form واطبع ما يلي في إطار Immediate:

- اطبع Set frm = Forms!Employees واضغط Enter يتم إنشاء متغير Variant المسمى frm ويشير إلى نموذج Employees.
- اطبع ?VarType(frm) واضغط Enter يتم عرض القيمة ٩ لأن المتغير يشير إلى كائن.
- اطبع ?IsObject(frm) واضغط Enter يتم عرض القيمة True لأن المتغير يشير إلى كائن.
- اطبع Set frm = Nothing واضغط Enter لا يشير المتغير بعد ذلك إلى أي كائن.
- اطبع ?IsObject(frm) واضغط Enter يتم عرض القيمة True لأن المتغير سبق وأن أشار مرة إلى كائن.
- اطبع Set frm = Empty واضغط Enter يتم توليد خطأ نوع ناتج عن عدم الملائمة لأنك لا تستطيع تعيين متغير كائن في Empty.
- اطبع Set frm = Null واضغط Enter يتم توليد خطأ نوع ناتج عن الملائمة لأنك لا تستطيع تعيين متغير كائن في Null.

الإجراءات والوحدات

الإجراء هو تتابع عبارات ينفذ كوحدة والإجراءات هي طوبة البناء الأساسية في VBA. يقوم الإجراء بمهمة محددة ويمكن أن تكون هذه المهمة عملية واحدة مثل فتح نموذج أو تشغيل استعلام لتحديد وعرض مجموعة من السجلات أو مجموعة معقدة من عمليات عديدة مثل استيراد واستغلال البيانات. يمكن للإجراء تقبل معلومات إضافية كوسائط تحديد طريقة عمل الإجراء.

أنواع الإجراءات

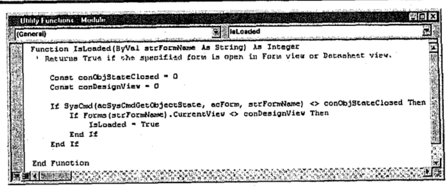
هناك ثلاثة أنواع للإجراءات في VBA وهي إجراءات الوظيفة والفرع والخاصية. ولكل نوع غرض مختلف وزوج من العبارات المخصوصة لتحدد البداية والنهاية.

إجراءات الوظيفة

يقوم إجراء الوظيفة بأداء مهمة ويمكن أن يرجع قيمة واحدة يتم حساب القيمة المرجعة في الإجراء وتعيينها الاسم الوظيفة كواحدة من عبارات الإجراءات. تستخدم إجراءات الوظيفة لإنشاء التخصيص الخاصة بك واستخدامها في التعبيرات بنفس الطريقة التي تستخدم بها الوظائف المضمنة تبدأ إجراء وظيفة بعبارة Function وتنتهي بعبارة End Function. بناء الجملة الأساسي لإجراء وظيفة هو.

```
Function functionname [(argumentlist)]
    [statements]
    [functionname = expression]
End Function
```

يوضح شكل ٧-١ إجراء وظيفة Northwind في IsLoaded تأخذ وظيفة IsLoaded اسم نموذج كوسيطة وتحدد إذا كان النموذج مفتوحاً وتقوم بتعيين اسم الوظيفة إلى True إذا كان النموذج مفتوحاً.

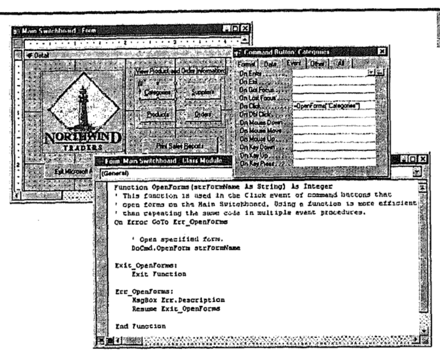


الشكل ٧-١ تحدد IsLoaded ما إذا كان النموذج مفتوحاً في طريقة عرض Form أو Datasheet.

يمكنك أيضاً استخدام إجراء وظيفة للتجاوب مع حدث. يقوم أكسس بتشغيل الإجراء تلقائياً عندما يحدث الحدث وينبذ قيمة الوظيفة المعادة إن وجدت يوضح الشكل ٧-٢ وظيفة OpenForms المستخدمة لفتح نموذج محدد عن طريق نقر زر أمر في نموذج Main Switchboard في Northwind تأخذ وظيفة OpenForms اسم نموذج كوسيلة وتفتح النموذج وتنهي دون إرجاع قيمة.

الشكل ٧-٢

تم تعيين إجراء وظيفة Open Forms لحدث Click الخاص بزر Categories أمر في Main Switchboard.



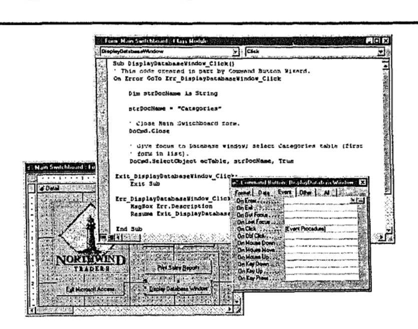
الإجراءات الفرعية

يقوم الإجراء الفرعي بأداء مهمة لا يرجع قيمة. بداية تستخدم الإجراءات الفرعية للتجاوب مع الأحداث. تبدأ الإجراء الفرعي بعبارة sub وتنتهيه بعبارة End Sub بناء الجملة الأساس للإجراء الفرعي هو

```
Sub subname [(argumentlist)]
    [statements]
End Sub
```

عندما تستخدم إجراء فرعي للتجاوب مع حدث فإنه يدعى إجراءات حدث يجب على إجراء الحدث اتباع قواعد محددة مثل قواعد التسمية. يتم تسمية إجراء حدث لحدث تم التعرف عليه من قبل نموذج أو تقرير كما يلي Form_eventname أو Report_eventname بحيث يكون eventname اسم الحدث. يتم تسمية إجراء حدث تم التعرف عليه من قبل أو تحكم في نموذج أو تقرير كما يلي: sectionname_eventname أو controlname_eventname.
بالإضافة إلى ذلك يتم تعريف قائمة وسيطة إجراء حدث لكل حدث مسبقاً. على سبيل المثال، إجراء حدث لحدث Click ليس له وسائط يوضح شكل ٧-٣ إجراء حدث DisplayDatabaseWindow_Click. يغلق إجراء الحدث هذا نموذج Main Switchboard ويعرض إطار Database وينهي.

الشكل ٧-٣



إجراء فرعي DisplayDatabaseWindow_Click "ج" تم تعيينه لحدث Click الموجود في زر أمر DisplayDatabaseWindow "ب" في نموذج Main Switchboard "أ".

إجراءات الخاصية

يؤدي إجراء الخاصية مهمة إنشاء خاصية تخصيص لكائن ما. في واقع الأمر، تقوم بإنشاء خاصية لها زوجين من إجراءات الخاصية باستخدام نوع واحد لتعيين الخاصية وأخر لإرجاع التعيين. تبدأ إجراء خاصية لتعيين الخاصية إما بعبارة Property Let عندما تقوم بتعيين قيمة أو بعبارة Property Set عندما تقوم بتعيين مرجع لكائن ما. لا تقوم إجراءات خاصية Property Let وSet بإرجاع قيم. تبدأ إجراء خاصية لإعادة التعيين بعبارة Property Get وتنتهي إجراء خاصية بعبارة End Property سنناقش إجراءات الخاصية في الفصل ١٤.

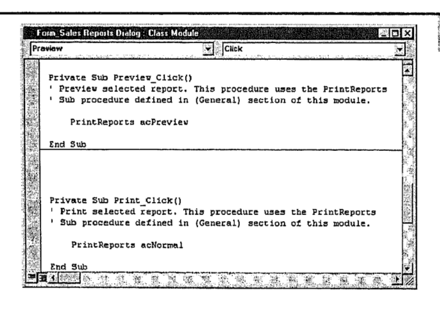
استدعاءات الإجراء

عند تطلب من أكسس تنفيذ إجراء ما، فأنت تستدعي الإجراء. يمكنك تصنيف الإجراءات تبعاً لكيفية استدعائك لها كما يلي:

في التعبيرات وإعدادات الخاصية لأن إجراءات الوظيفة تقوم بإرجاع قيم، يمكنك استدعائها مباشرة في التعبيرات وإعدادات الخاصية بنفس الطريقة التي تستدعي بها الوظائف المضمنة. يمكنك استخدام إجراءات الوظيفة كإعدادات في أوراق خاصية الجداول والاستعلامات والنماذج والتقارير وأيضاً خلايا الحالة وإعدادات الوسيطة في الماكروات وفي خلايا الحقل والمعايير في الاستعلامات.

كإجراءات تتعامل مع الحدث يمكنك استخدام كلاً من إجراءات الوظيفة الفرعية للتجاوب مع الأحداث. يدعى الإجراء الذي تعنيه لحدث إجراء يتعامل مع حدث يحفظ مايكروسوفت المصطلح إجراء حدث لإجراء فرعي تم تعيينه لحدث ما.

كإجراءات دعم أو إجراءات عامة تستدعي الإجراءات بطريقة مباشـــرة عندمـــا تقــوم بتضمينها في عبارة إجراء أخر يتم الإشارة إلى الإجراء الذي تستدعيه من إجراء أخـــر كإجراء دعم أو إجراء عام تستخدم إجراءات الدعم لتقسيم مهمة معقدة بعمليات متعـــددة إلى مجموعة من الإجراءات الأكثر بساطة ولإنشاء إجراءات عامـــة الغـــرض يمكـــن استدعاؤها من إجراءات أخرى عديدة. يمكنك استخدام كلاً مـــن إجـــراءات الوظيفـــة والإجراءات الفرعية كإجراءات دعم، قم باستخدام إجراء وظيفة عندما تريد تمرير قيمة من إجراء إلى أخر وإجراء فرعي في غير ذلك. على سبيل المثال، تقـــوم إجـــراءات الحدث الخاصة فـــي Print Preview وأزرار Print فـــي نمـــوذج Sales Report Dialog في Northwind باستدعاء الإجراء الفرعي PrintReports انظر شـكل ٧-٤ عندما يقوم إجراء باستدعاء إجراء دعم، يقاطع أكسس تنفيذ الإجراء المستدعي وينفـــذ إجراء الدعم ثم يرجع لاستكمال تنفيذ الإجراء المستدعي.



الشكل ٧-٤

يستـــدعي إجـــراء Preview-Click وClick Print الإجراء الفرعـــي .Print Reports

الوحدات

الوحدة هي الكائن الذي تستخدمه لإنشاء وتخزين الإجراءات. الوحدة هي شفرة تسرد إجـــراء أو أكثر أو إعلانات أو عبارات تقوم بتخزينها أنت سوياً كوحدة. هناك نوعان من الوحـــدات التـــي تستخدمها لتنظيم إجراءاتك وهي وحدات الفئة والوحدات القياسية.

وحدات الفئة

تحتوي وحدة الفئة على إجراءات تستخدمها لإنشاء تعريف لكائن جديد. تتضمن الإجراءات فـــي الوحدة الخصائص والطرق الخاصة بالكائن الجديد. هناك نوعان من وحدات الفئة وهما هـــؤلاء

المرتبطون بالنماذج والتقارير والآخرون الذين يتواجدون بصورة مستقلة عن النماذج والتقارير "وحدات الفئة المستقلة موضوع متقدم لم يتم بحثه بصورة وافية في هذا الكتاب".

تقوم بإنشاء وحدة نموذج أو تقرير عن طريق فتح النموذج أو التقرير في رؤية Design واختيار View ⇦ Code أو نقر زر Code في شريط الأدوات يفتح إطار Module ويعرض وحدة سيتم تخزينها كجزء من النموذج أو التقرير عندما تحفظ النموذج أو التقرير. يقوم أكسس تلقائياً بتعيين خاصية HasModule الخاصة بالنموذج أو التقرير في Yes ويسمى وحدة النموذج أو التقرير المضمنة باستخدام اسم النموذج أو التقرير كما يلي Form_formname أو Report_reportname. يتم تخزين هذه الوحدات المضمنة مع النموذج أو التقرير ولا يتم سردها منفصلة في إطار Database.

الغرض الأساسي من وحدة النموذج أو التقرير هو تخزين كل إجراءات الحدث للأحداث التي تم التعرف عليها بواسطة النموذج أو التقرير والتحكمات الأقسام الخاصة بهما. تستطيع وحدة النموذج أو التقرير تخزين إجراءات الوظيفة والإجراءات الفرعية العامة، وحدات النموذج والتقرير هي طرق مناسبة لتخزين الإجراءات المرتبطة بنموذج أو تقرير عندما تنسخ النموذج أو التقرير فأنت أيضاً تنسخ وحدة النموذج أو التقرير عندما تلغي نموذجاً أو تقريرا، فأنت تلغي وحدة النموذج أو التقرير أيضاً. تستطيع إلغاء وحدة النموذج أو التقرير دون إلغاء النموذج أو التقرير في No.

الوحدات القياسية

يتم سرد كل وحدة قياسية ككائن منفصل في لوح Modules في إطار Database. قواعد التسمية للوحدة القياسية هي نفس القواعد الخاصة بأي كائن إطار Database. غالباً، تستخدم الوحدات القياسية لحمل وظائف التخصيص وإجراءات الدعم التي تريد استدعاءها من أي مكان في التطبيق.

عرض الوحدة

كل الوحدات لها نفس الغرض تبدأ الوحدة بقسم Declarations يحتوي على إعدادات خيار وإعلانات تطبق على كل إجراء في الوحدة تستخدم قسم Declarations لتعريف أنواع بيانات التخصيص ولإعلان المتغيرات والثوابت التي تريد مشاركتها بين الإجراءات في الوحدة وبين الإجراءات في الوحدات الأخرى وحتى في قواعد البيانات الأخرى بالإضافة إلى ذلك إذا كانت تستخدم. إجراءات تم تخزينها خارجياً في ملف منفصل يدعى مكتبة ربط حيوية (DLL) فأنت توفر معلومات عن اسم الإجراء والمكتبة في قسم Declarations وتدعى الشفرة التي تدخلها في قسم Declarations شفرة مستوى الوحدة.

يتبع قسم الوحدة Declarations قسم Procedures الذي تخزن فيه إجراءات الوحدة يجب أن تكون للإجراءات الموجودة في الوحدة أسماء متميزة مع ذلك يمكن أن تكون للإجراءات في وحدات مختلفة نفس الاسم.

مراجع الإجراء

بناء الجملة للإشارة إلى إجراء في وحدة هو:

modulename.procedurename

وهذا البناء هو مرجع مؤهل تماماً للإجراء. على سبيل المثال، يشير ما يلي إلى الإجراءات في Northwind:

[Utility Function].IsLoaded

[Form_Main Switchboard].OpenForms

في المثالين كليهما، يجب أن تضمن أسماء الوحدة في أقواس مربعة لأن أسماء الوحدات تحتوي على مسافات غالباً، تستطيع حذف اسم الوحدة وتشير إلى الإجراء عن طريق اسمه. القواعد الأساسية هي كما يلي:

- عندما تشير إلى إجراء من داخل نفس الوحدة يمكنك حذف المرجع لاسم الوحدة لأن VBA يفترض أنك تشير إلى نفس الوحدة عندما تحذف اسم الوحدة كاستثناء، إذا كان الإجراء له نفس الاسم كالوحدة، يجب أن تضمن اسم الوحدة.
- عندما تكون في نموذج أو تقرير، يمكنك في أغلب الأوقات الإشارة إلى إجراء في وحدة النموذج أو التقرير عن طريقة اسمه لأن VBA يفترض أنك تشير إلى وحدة النموذج عندما تحذف اسم الوحدة كاستثناء، إذا كان للنموذج أو التقرير نفس الاسم مثل الإجراء، يجب أن تضمن اسم الوحدة.
- عندما تشير إلى إجراء في وحدة نموذج من خارج الوحدة، يجب أن تضمن اسم الوحدة.
- عندما تشير إلى إجراء له نفس الاسم في وحدتين أو أكثر أخرتين، يجب أن تضمن اسم الوحدة في المرجع بحيث يعلم VBA أي الإجراءات تشير إليها.

الأسماء في فيجوال بيسيك

يجب أن تبدأ أسماء الوحدات والإجراءات والمتغيرات والثوابت بحرف ولا يمكنها الاحتواء على مسافات مضمنة أو أحرف نوع الإعلان ولا يمكن أن تكون كلمات أساسية مفيدة يستخدمها VBA كجزء من لغته. من أمثلة الكلمات الأساسية المفيدة If و Then و End و While و loop و Sub.

يجب أن لا يتعارض اسم الوحدة مع مصطلح التسمية الذي يستخدمه أكسس لوحدات النموذج والتقرير لا تبدأ اسم وحدة مثل Form أو Report.

يمكن أن يكون للإجراء نفس الاسم مثل الوحدة في هذه الحالة يجب أن تستخدم المرجع المؤهل كليا للإجراء.

الإجراءات العامة ضد الإجراءات الخاصة

عندما تقوم بإنشاء إجراء، يمكنك تحديد ما إذا كان الإجراء عام أم خاص يمكن تشغيل الإجراء العام بواسطة إجراءات في وحدات أخرى ويمكن تشغيل إجراء خاص فقط بواسطة إجراءات في نفس الوحدة.

يمكنك استخدام الكلمات الأساسية Public وPrivate لتحديد توفر الإجراء للإجراءات الأخرى وهذا يدعى المرئي أو المجال فرضياً، الإجراءات التي ليس لها من الكلمات الأساسية السابقة في عبارات الإعلان الخاصة بها هي إجراءات عامة ويمكن استدعاؤها عن طريق أي إجراء في التطبيق. لا تحتاج لاستخدام الكلمة الأساسية Public لكن ستكون الشفرة أسهل في الفهم إذا استخدمت الكلمة الأساسية Public في عبارة إعلان الإجراء العام. على سبيل المثال، تحدد Public Function FirstFunction إجراء FirstFunction كإجراء عام.

عندما تستخدم أدوات بناء الشفرة المضمنة لإجراء حدث، يقوم أكسس بتضمين كلمة Private الأساسية في قالب الشفرة فرضياً لأن بطبيعة الحال، تستخدم إجراءات الحدث فقط في النموذج أو التقرير اللذين تخزن فيها. مع ذلك يمكنك إلغاء الكلمة الأساسية Private لجعل إجراء حدث عام.

تلميح اجعلها قاعدة لك استخدام الكلمة الأساسية Public أو Private عند إنشاء الإجراءات. ستتعلم بسرعة عن المجال والذي هو واحد من أعقد وأصعب مفاهيم برمجة VBA.

الإجراءات العامة المخزنة في الوحدات القياسية متوفرة أيضاً لقواعد البيانات الأخرى فرضياً. يمكن تقييد الإجراءات العامة في وحدة قياسية للاستخدام في قاعدة البيانات الحالية فقط عن طريق تضمين Option Private مع إعدادات الخيار الأخرى في قسم Declarations الخاص بالوحدة. على العكس، الإجراءات العامة المخزنة في وحدات النموذج والتقرير متوفرة فقط في قاعدة البيانات الحالية.

يمكنك الإشارة لإجراء عام في وحدة نموذج أو نموذج فرعي عندما يكون النموذج مفتوحاً في طريقة عرض Form كطريقة للنموذج المعروض في تحكم النموذج الفرعي. استخدم المرجع

Forms!formname!subformcontrolname.Form.procedurename لاستخدام الإجراءات العامة في وحدة قياسية والتي تم تخزينها في قاعدة معلومات أخرى، يجب عليك أولاً إضافة مرجع لقاعدة البيانات الأخرى. انظر الفصل ١٥ لمعلومات من إضافة المرجع.

أمثلة بسيطة للإجراءات

لنبدأ استكشافنا للإجراءات بإنشاء واستدعاء بعض إجراءات الوظيفة والإجراءات الفرعية البسيطة. الإجراءات التي تقوم بإنشائها الآن تقوم بأكثر من عرض رسائل التخصيص باستخدام وظيفة MsgBox لتحديد أنها يتم تنفيذها بعد تقديم الآليات في هذا الفصل والفصلين التاليين، سنتحدث عن الإجراءات الأكثر تعقيداً والضرورية لجعل قاعدة المعلومات تلقائية. أيضاً في القسم التالي من هذا الفصل، سنجري دراسة مدققة حول إطار Design الخاص بالوحدة.

إنشاء إجراء حدث

لأن إجراءات الحدث شائعة جداً في تطبيق أكسس يوفر أكسس أدوات بناء شفرة إجراء حدث خاصة لإنشاء إجراء حدث باستخدام الأدوات المضمنة في طريقة عرض Design حدد الكائن الذي يتعرف على الحدث وانقر خاصية الحدث في ورقة الخاصية ثم انقر زر Build أيمن مربع الخاصية، حدد Code Builder وانقر OK.

يتم عرض إطار Module بوحدة النموذج أو التقرير المضمنين. يعرض VBA تلقائياً سطرين من الشفرة يدعيان قالب شفرة أو (stub) ويضع نقطة الإدراج بينهما جاهزة حتى تدخل شفرة الإجراء يشير السطر الأول من قالب الشفرة إلى بداية إجراء الحدث ويعتمد على الحدث الذي قمت بتحديده يقوم أكسس تلقائياً بتسمية إجراءات الحدث ويحدد الوسائط إذا قمت بتحديد حدث تم التعرف عليه من قبل النموذج أو التقرير، سيكون السطر الأول واحداً مما يلي:

```
Private Sub Form_eventname(eventargumentlist)
Private Sub Report_eventname(eventargumentlist)
```

إذا حددت حدثاً تم التعرف عليه عن طريق تحكم أو قسم، سيكون السطر الأول:

```
Private Sub objectname_eventname(eventargumentlist)
```

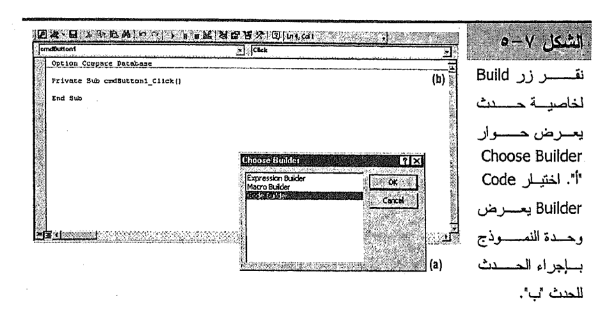
يشير السطر الثاني إلى نهاية الإجراء وهو كذلك في كل الإجراءات الفرعية.

سنقوم بإنشاء الإجراء الأول كإجراء حدث يعمل عندما يتم نقر زر أمر:

١ ـ أنشئ قاعدة معلومات جديدة وسميها Ch7_Examples.

١ ـ أنشئ نموذجاً جديداً يدعى frm1. ضع زر أمر في النموذج وعين خاصية Name في cmdButton1 وخاصية Caption في Call an Event Procedure.

٣ـ انقر في خاصية حدث الزر OnClick. انقر زر Build أيمن مربـــع الخاصيـــة، حـــدد Code Builder في حوار Choose Builder، انظر شكل "٧-٥ أ" ثم انقر OK يفتـــح إطار Module لوحدة نموذج مضمنة انظر شكل "٧-٥ ب".



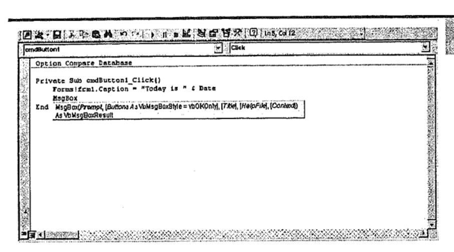
الشكل ٧-٥ نقـــر زر Build لخاصيـــة حـــدث يعـــرض حـــوار Choose Builder "أ". اختيـــار Code Builder يعـــرض وحـــدة النمـــوذج بـــإجراء الحـــدث للحدث "ب".

تبدأ الوحدة بقسم Declarations. يعرض قسم Declarations عبارة خيار يتضمنهـــا VBA فرضياً: Option Compare Database تستخـــدم خيـــار Option Compare Database لتحديد أنك تريد أكسس أن يقارن تعبيرات السلسلة تبعاً لترتيب صنف قاعدة المعلومات الذي تـــم إعداده لقاعدة المعلومات يفصل السطر أسفل عبارة Option والذي يدعى فاصل إجـــراء، قســـم Declarations عن الإجراء الأول يتبع قسم Declarations سطرين لقالب الشفرة وتوجد نقطـــة الإدراج بينهم.

٤ـ أنشئ إجراء الحدث بطبع السطرين الخاصين بالشفرة بين أسطر قالب الشفرة كما هـــو موضح بالأسفل يستخدم السطر الأول عبارة تعيين خاصية النموذج Caption للبيانـــات الحالية باستخدام وظيفة Date المضمنة. يســـتخدم الســـطر الثـــاني وظيفـــة MsgBox المضمنة لعرض مربع رسالة تخصيص وبينما تقوم بالطبع، يقوم VBA بتدقيق إدخالاتك ويجرى تصحيحات واقتراحات. على سبيل المثال، بعد أن تطبـــع MsgBox. يعـــرض VBA مربعاً ببناء الجملة للوظيفة كدليل للوسائط المطلوبة والخيارية وللقيمة المرجعـــة انظر شكل ٧-٦.

```
Private Sub cmdButton1_Click()
    Forms!frm1.Caption = "Today is " & Date
    MsgBox "This is cmdButton1_Click in module Form_frm1"
End Sub
```

ملاحظة إذا أدخلت الأقواس بعد وظيفة Date، Date() يزيل VBA الأقواس تلقائياً عندما تستخدم وظيفة Date في إعداد خاصية، أو تغيير استعلام أو ماكرو يجب أن تضمن أقواساً بعد الوظيفة، مع ذلك لا تستخدم الأقواس في برمجة VBA ويقوم VBA بإزالة الأقواس.



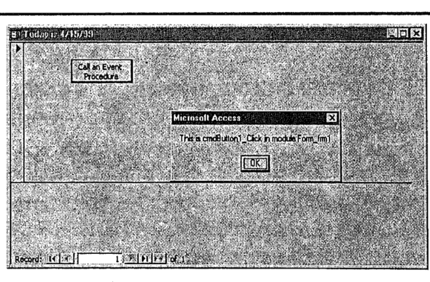

```
Option Compare Database

Private Sub cmdButton1_Click()
    Forms!frm1.Caption = "Today is " & Date
    MsgBox
End  MsgBox(Prompt, [Buttons As VbMsgBoxStyle = vbOKOnly], [Title], [HelpFile], [Context])
     As VbMsgBoxResult
```

الشكل ٧-٦ عندما تقوم بطبع وظيفة مضمنة، يعرض VBA مربع بناء الجملة كدليل لوسائط الوظيفة والقيمة المرجعة.

٥- انقر في النموذج. تعرض خاصية OnClick لزر الأمر الإعداد (Event Procedure) لتحديد أن إجراء الحدث قد تم تعيينه تلقائياً لحدث Click الخاص بالزر.

٥- بدل لطريقة عرض Form وانقر زر الأمر. يقوم الإجراء بتعيين تعليق النموذج للتاريخ الحالي ويعرض مربع الرسالة انظر شكل ٧-٧. انقر OK لإغلاق مربع الرسالة. احفظ النموذج.



الشكل ٧-٧ يغير إجراء الحدث تعليق النموذج ويعرض مربع رسالة.

مقارنات السلسلة

فرضيا، العبارة الأولى في قسم Declarations الخاص بوحدة ما هي عبارة Option Compare Database والتي تشير إلى أن VBA يستخدم نفس ترتيب صنف للمقارنة تعبيرات السلة في الوحدة التي يستعملها أكسس للمقارنة بين تعبيرات السلسلة في قاعدة البيانات يمكنك تغيير الأبجدية المستخدمة لمقارنات السلسلة في جدولة General في حوار Options. تعرض قائمة مربع التحرير والسرد الخاصة في New Database Sort Order ترتيبات الصنف المتوفرة.

تمثل قيمة General الفرضية ترتيب صنف غير حساس للحالة الخاصة بالأحرف معتمدا على الأبجدية الإنجليزية. إذا غيرت الإعداد، يتغير ترتيب الصنف فقط لقواعد بيانات جديدة، ولقواعد بيانات موجودة تتضمن قاعدة البيانات الحالية التي لا تتأثر.

بناء الجملة لعبارة Option Compare هو:

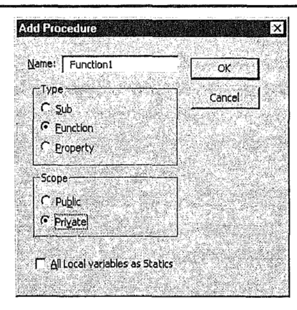
Option Compare {Binary| Text| Database}

يؤدي استخدام Binary إلى مقارنات سلسلة مصنوعا تبعا لترتيب صنف تحسس لحالة الأحرف يعتمد على شفرة ASCII للأحرف. يؤدي استخدام Text إلى مقارنات سلسلة لا تحسس لحالة الأحرف معتمدة على شفرة ASCII بطبيعة الأمر، تستخدم الفرض، إذا لم تضمن عبارة Option Compare VBA يستخدم Option Compare Binary.

إنشاء إجراء وظيفة

بعد ذلك، ستقوم بإنشاء إجراء وظيفة وتستدعي وظيفة في تعبير لخاصية ControlSource لتحكم محسوب. سنعرض القيمة المرجعة من قبل الوظيفة في مربع نص في النموذج.

١ـ بدل إلى طريقة عرض Design وانقر Code أو اختر View ⇦ Code اختر Insert ⇦ Procedure أدخل Function1 في مربع Name في حوار Add Procedure، اختر Function في مجموعة خيار Type انقر Private في مجموعة خيار Scope انظر شكل ٧-٨ وانقر OK. يدرج VBA خط إجراء فاصل تحت الإجراء السابق ويعرض قالب الشفرة للوظيفة.



الشكل ٧-٨ تستخدم حوار Add Procedure لتسمية الإجراء الجديد ولتحديد نوعه ومجاله.

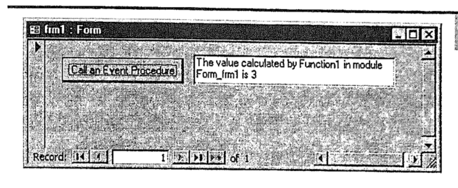
١ـ أدخل إجراء Function1 أسفل لإجراء الوظيفة هذا وسيطة تدعى A والتي يجب دعمها عندما تشغل الإجراء. يضيف السطر الأول من الشفرة واحد للوسيطة ويحدد المجموع كقيمة إرجاع الوظيفة يعرض السطر الثاني رسالة.

```
Private Function Function1(A)
  Function1 = A + 1
  MsgBox "This is Function1 in module Form_frm1"
End Function
```

٣ـ ضع مربع نص في النموذج عين خاصية Name في txtCalculated واطبع التعبير التالي في خاصية ControlSource:

```
= "The value calculated by Function1 in module Form_frm1 is " &
Function1(2)
```

٤- بدل لطريقة عرض Form وانقر OK عندما يتم عرض مربع الرسالة يعرض مربع النص النتيجة انظر شكل ٧-٩ احفظ النموذج.



الشكل ٧-٩

يتضمن تعبير خاصية ControlSource وظيفة التخصيص.

عندما تعرض النموذج في طريقة عرض Form، يقيم VBA تعبير ControlSource كما يلي: يشتغل إجراء الوظيفة بوجود الوسيطة الخاصة به في ٣، ترجع الوظيفة القيمة ٤، ثم يستخدم VBA القيمة المرجعة في تعبير ControlSource ويعرض النتيجة في مربع النص.

تشغيل إجراء أخر في الوحدة

في VBA، تستطيع استدعاء إجراء من إجراء أخر. هذا يعني أنه باستطاعتك تقسيم مهمة برمجة معقدة إلى مجموعة من المهام الأكثر سهولة وإنشاء إجراءات منفصلة لكل مهمة تحصل على فائدتين مهمتين من تقسيم المهام هذا وهما:

- إجراءات أكثر لمهام أكثر بساطة في الكتابة والتصحيح.
- عندما توجد مهمة تحتاج لاستخدامها مرة أخرى في جزء أخر من التطبيق أو في تطبيق أخر، يمكنك كتابة الإجراء بحيث تعيد استخدامه دون تعديل.

السبب الثاني من الأهمية بمكان بحيث يجب عليك اتباعه كمرشد برمجة أساس لذلك عليك إنشاء إجراءات قابلة للاستخدام مرة أخرى كلما سمح الأمر.

سنقوم بإنشاء زر أمر يستدعي إجراء Function1:

١- بدل لطريقة عرض Design، ضع زر أمر في النموذج وعين خاصية Name الخاصة به في cmdButton2 وخاصية Caption إلى Call a procedure in the form's module.

١- انقر خاصية حدث الزر OnClick. انقر زر Build يمين مربع الخاصية، حدد Code Builder في حوار Choose Builder وانقر OK. يفتح إطار Module ويعرض قالب الشفرة لإجراء الحدث الجديد.

٣ـ اطبع إجراء الحدث أسفل. يستدعي السطر الأول من الشفرة إجراء Function1 بالوسيطة ٤. تتم الإشارة للإجراء الذي تم استدعاؤه procedure called أو nested procedure يعرض السطر الثاني رسالة:

```
Private Sub cmdButton2_Click ()
    Call Function1(3)
    MsgBox "This is cmdButton2_Click in Form_frm1"
End Sub
```

٤ـ بدل لطريقة عرض Form. يقوم التعبير في خاصية مربع النص ControlSource باستدعاء Function1 بواسطة ٣ كما سبق. انقر زر OK حتى يستكمل VBA تقييم تعبير ControlSource.

٥ـ انقر زر الأمر الجديد بتعليق Call استدع إجراء وحدة النموذج يستدعي إجراء الحدث Function1 مع ٤ كوسيطة يضيف إجراء Function1 واحد للوسيطة ويرجع ٥ كقيمة للوظيفة. مع ذلك، عندما تستخدم عبارة Call لاستدعاء وظيفة، يتجاهل VBA قيمة الوظيفة المرجعة. يعرض Function1 الرسالة الخاصة به عندما تنقر OK، يعرض إجراء حدث cmdButton2_Click الرسالة.احفظ النموذج.

تشغيل إجراء في وحدة أخرى

تستطيع أيضاً تشغيل إجراء مخزن في وحدة أخرى. في معظم قواعد البيانات، تقوم بإنشاء وحدة قياسية واحدة أو أكثر وتستخدمها لتخزين الإجراءات العامة التي تستخدمها في نماذج أو تقارير متعددة ستقوم بإنشاء وحدة قياسية وتستخدمها لتخزين إجراء وظيفة جديدة سيشتغل من زر ثالث في النموذج.



١ـ انقر الزر المنسدل New Object في شريط الأدوات واختر Module. يفتح إطار Module عارضا قسم Declarations في الوحدة القياسية يعرض قسم Declarations عبارة الخيار الفرضية. احفظ الوحدة الجديدة على شكل bas1.

١ـ قم بإنشاء إجراء وظيفة جديد بطبع Public Function Function1(A) على سطر جديد. تستخدم الكلمة الأساسية Public لجعل الإجراء متوفر لإجراءات في وحدات أخرى عندما تضغط Enter يدخل VBA تلقائياً فاصل إجراء ويعرض قالب الشفرة أدخل الإجراء الموضح فيما يلي. "تستخدم نفس الاسم للوظيفة الجديدة تعمداً".

```
Public Function Function1(A)
   Function1 = A + 2
   MsgBox "This is Function1 in module bas1."
End Function
```

٣- انقر في النموذج وبدل لطريقة عرض Design قم بإنشاء زر أمر جديد وعين خاصية Name في cmdButton3 وخاصية Caption إلى Call a procedure in another module قم بإنشاء إجراء الحدث التالي لحدث الزر Click لأن هناك إجراءات Function1، سيقوم VBA بالاختيار.

```
Private Sub cmdButton3_Click()
   Call Function1(3)
   MsgBox "This is cmdButton3_Click in module Form_frm1."
End Sub
```

٤- بدل لطريقة عرض Form. كما سبق، يقوم التعبير في خاصية مربع النص ControlSource باستدعاء Function1 بوسيطة ٣ انقر زر OK.

٥- انقر الزر الجديد. تشير مربع الرسالة إلى الاختيار الذي قام به VBA، يشغل Function1 في وحدة النموذج. عندما يستدعي الإجراء إجراء ثاني، يبحث VBA عن الإجراء الذي تم استدعاؤه في نفس الوحدة أولاً وإن لم يجد الإجراء فإنه يبحث عنه في الوحدات الأخرى في هذه الحالة، وجد VBA إجراء Function1 في وحدة النموذج وقام بتشغيله، لاستدعاء إجراء Function1 في الوحدة القياسية، يجب أن تضمن مرجع لاسم الوحدة (المرجع المؤهل عملية للإجراء).

٥- انقر في النموذج وعدل إجراء الحدث كما هو موضح كما يلي:

```
Private Sub cmdButton3_Click()
   Call bas1.Function1(3)
   MsgBox "This is cmdButton3_Click in module Form_frm1."
End Sub
```

٦- انقر الزر الجديد هذه المرة يشغل VBA إجراء Function1 في الوحدة القياسية احفظ النموذج.

إخفاء إجراء

عندما تستخدم أدوات بناء شفرة إجراء الحدث المضمنة يتضمن قالب الشفرة لإجراء الحدث الكلمة الأساسية Private في VBA، تقرر أي الإجراءات في التطبيق الخاص بك يمكنها استخدام إجراء معين، يمكنك جعل الإجراء متوفراً فقط لإجراءات أخرى مخزنة في نفس الوحدة وإخفاء

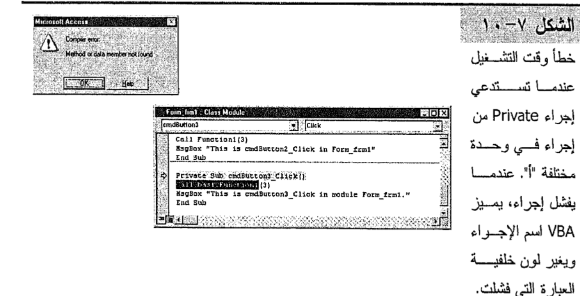
إجراءات نموذج الإجراء المخزنة في الوحدات الأخرى باستخدام الكلمة الأساسية Private في السطر الأول من عبارة إعلان الإجراء:

١- انقر في إجراء Function1 للوحدة bas1 وغير Public إلى Private في السطر الأول من عبارة إعلان Function1. بهذا الإعلان، يكون الإجراء متوفراً فقط لإجراءات أخرى في وحدة bas1.

٢- انقر في النموذج ثم انقر الزر الجديد لا يستطيع أكسس تشغيل الإجراء ويعرض رسالة خطأ موضحة في شكل "٧-١٠ أ" عندما تستخدم الكلمة الأساسية Private لإخفاء الإجراء. لا يكون الإجراء مرئياً للإجراءات الموجودة في الوحدات الأخرى من وجهة نظر إجراء في وحدة النموذج. إجراء Function1 في bas1 لا يكون موجوداً، لذلك يتم توليد خطأ وقت تشغيل.

٣- انقر OK يعرض إطار Module حلولاً لهذه المشكلة يميز VBA اسم الإجراء الذي فشل ويعرض سهماً في الهامش الأيسر ويغير لون خلفية عبارة الشفرة التي فشلت. انظر شكل "٧-١٠ ب".

٤- انقر مربع Close لإطار Module انقر Yes لإعادة تعيين الشفرة وإيقاف تشغيل للإجراء.




```
Call Function1(3)
MsgBox "This is cmdButton2_Click in Form_frm1"
End Sub

Private Sub cmdButton3_Click()
Call bas1.Function1(3)
MsgBox "This is cmdButton3_Click in module Form_frm1."
End Sub
```

الشكل ٧-١٠
خطأ وقت التشغيل عندما تستدعي إجراء Private من إجراء في وحدة مختلفة "أ". عندما يفشل إجراء، يميز VBA اسم الإجراء ويغير لون خلفية العبارة التي فشلت.

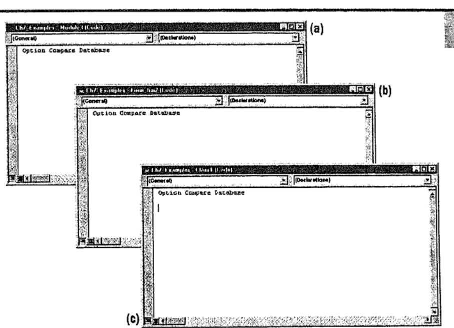
الأمثلة التي تناولناها حتى الآن سهلة لأنها لا تستخدم المتغيرات "باستثناء الوسيطة لإجراء الوظيفة" يشرح باقي هذا الفصل بتوسيع أكثر آليات إنشاء وتشغيل الإجراءات.

بيئة برمجة أكسس VBA

سنبدأ دراستنا لبيئة برمجة أكسس VBA باستقصاء لطريقة عرض Module وآليات إنشاء وتحرير الشفرة.

عرض وحدة في طريقة عرض Module

تقوم بإنشاء شفرة في إطار Module في طريقة عرض Module يمكنك فتح قياسي أو نموذج أو تقرير أو وحدة فئة مستقلة كما هو موضح في الأقسام التالية في كل حالة، يفتح إطار Module عارضاً قسم Declarations بالخيار الفرضي يشير شريط عنوان الإطار إلى اسم ونوع الوحدة. انظر شكل ٧-١١.



الشكل ٧-١١
إطار Module لوحدة قياسية "أ" ووحدة نموذج "ب" ووحدة فئة مستقلة "ج" يعرض شريط العنوان اسم الوحدة ونوعها.

فتح وحدة قياسية

تفتح وحدة قياسية جديدة بنقر زر New في لوح Modules لإطار Database أو بنقر زر New Object في شريط الأدوات واختيار Module إذا كنت في طريقة عرض Module، تستطيع فتح وحدة قياسية جديدة باختيار Insert ⇦ Module أو عن طريق زر Insert Module في شريط الأدوات.

تفتح وحدة قياسية موجودة عن طريق تحديد الوحدة في إطار Database ونقرها مرتين أو عن طريق زر Design.

فتح وحدة نموذج أو تقرير

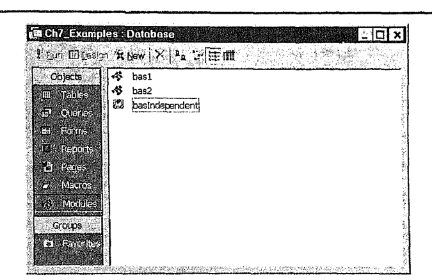
 تفتح وحدة نموذج أو تقرير عن طريق فتح النموذج أو التقرير في طريقة عرض Design واختيار View ⇦ Code من القائمة أو زر Code في شريط الأدوات يمكنك فتح نموذج أو تقرير والوحدة الخاصة بها على التوالي عن طريق تحديد النموذج أو التقرير في إطار Database ونقر زر Code في شريط الأدوات. فرضياً لا يحتوي التقرير أو النموذج الجديد على وحدة ويدعى النموذج أو التقرير lightweight أو حمل خفيف. يقوم أكسس بإنشاء وحدة التقرير أو النموذج في أول مرة تحاول فيها الوحدة.

يتم حفظ وحدة النموذج أو التقرير مع النموذج ولا تكون الوحدة موجودة في لوح Module لإطار Database.

فتح وحدة فئة مستقلة

تفتح وحدة فئة مستقلة عن طريق نقر زر New Object في شريط الأدوات واختيار Class Module. إذا كنت في طريقة عرض Design تستطيع فتح وحدة فئة مستقلة جديدة باختيار أمر Insert ⇦ Class Module في القائمة أو عن طريق نقر زر Class Module في شريط الأدوات.

عندما تحفظ وحدة فئة مستقلة، تكون موجودة داخل جدولة Modules لإطار Database مع الوحدات القياسية يستخدم VBA رموزاً مختلفة للوحدات القياسية والمستقلة انظر شكل ٧-١٢.



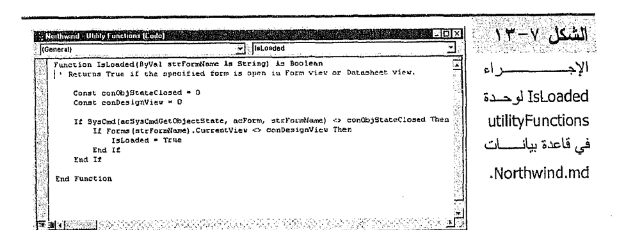
الشكل ٧-١٢ يتم سرد الوحدات القياسية ووحدات الفئة المستقلة برموز مختلفة في إطار Database.

تفتح وحدة فئة مستقلة موجودة عن طريق تحديد الوحدة في إطار Database ونقرها نقراً مزدوجاً أو عن طريق نقر زر Design.

تحديد محتويات وحدة

تستخدم مربعي التحرير والسرد أسفل شريط عنوان إطار Module لتحديد محتويات الوحدة.

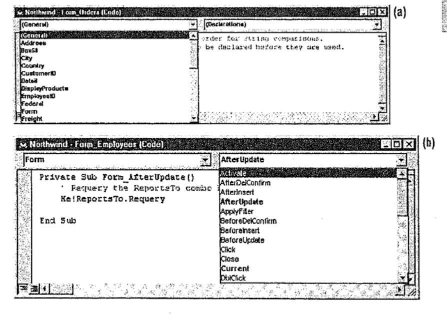
لوحدة قياسية، يعرض مربع التحرير والسرد في اليسار General كعنصر وحيد يمثل عنصر General الوحدة نفسها. يعرض مربع التحرير والسرد في اليمين Declarations كالعنصر الأول متبوعاً بأسماء الإجراءات العامة في الوحدة. يعرض شكل ٧-١٣ إجراء IsLoaded مخزنة في وحدة قياس. UtilityFunctions في نموذج تطبيق Northwind.mdb. يمثل عنصر Declarations قسم Declarations للوحدة عندما تحدد اسم إجراء من القائمة، يتم عرض شفرة الإجراء.



الشكل ٧-١٣ الإجراء IsLoaded لوحدة utilityFunctions في قاعدة بيانات Northwind.md.

لوحدة نموذج أو تقرير، يقوم مربع التحرير والسرد في اليسار بسرد (General) متبوعاً بقائمة بالكائنات المرتبطة بالنموذج أو التقرير بما فيها اسم كل تحكم يتعرف على الأحداث واسم كل قسم معروف والنموذج أو التقرير نفسه محدداً بواسطة Form أو Report. يوضح شكل ٧-١ قائمة الكائنات في نموذج Orders في قاعدة بيانات Northwind.mdb عندما تحدد عنصر "General"، يعرض مربع التحرير والسرد في اليمين "Declarations" كالعنصر الأول متبوعاً بأسماء الإجراءات العامة في الوحدة. عندما تحدد كائن من مربع التحرير والسرد في اليسار يعرض مربع التحرير في اليمين قائمة بالأحداث التي يتعرف عليها الكائن.

عندما تحتوي الوحدة على إجراء حدث لحدث كائن ما، يتم عرض هذا الحدث بخط أسود داكن. عندما تحدد حدثاً معروضاً بالخط الأسود الداكن، يتم عرض إجراء الحدث المقابل على سبيل المثال، يعرض شكل "٧-١٤ ب" إجراء حدث Form_AfterUpdate عندما تحدد حدثاً ليس معروضاً بالأسود الداكن، يقوم VBA بإنشاء وعرض قالب شفرة لإجراء الحدث. بمجرد إنشاء قالب الشفرة لحدث ما، تعرض القائمة الحدث بالأسود الداكن إذا لم تدخل أية شفرة بين سطور القالب بعد أن تلغي إجراء حدث أو قالب شفرة لحدث ما، تعرض القائمة الحدث. بالخط العادي.



الشكل ٧-١٤ توفر مربعات السرد والتحرير لإجراء وحدة نموذج أو تقرير قائمة جرد للوحدة يسرد مربع سرد وتحرير Object الكائنات في النموذج أو التقرير "أ" ويسرد مربع سرد وتحرير Procedures الأحداث التي تم التعرف عليها من قبل كائن تم تحديده "ب".

تلميح يمكنك التنقل خلال كل الإجراءات في وحدة باستخدام مفاتيح الاختصارات، فضغط Ctrl+Up Arrow أو Ctrl+Down Arrow يعرض الإجراء السابق أو التالي على التوالي.

عرض إجراء حدث مباشرة

طريقة أخرى لعرض إجراء حدث معين هي بنقر خاصية الحدث في ورقة خاصية كائن أخر. إذا تواجد إجراء حدث لحدث ما، يكون إعداد خاصية الحدث "Event Procedure"، يفتح نقر زر Button يمين مربع الخصائص وحدة النموذج أو التقرير ويعرض إجراء الحدث.

يمكنك إنشاء إجراء حدث جديد لحدث ما عن طريق نقر زر Build على يمين مربع خاصية الحدث واختيار Code Builder في حوار Code Builder. عندما تنقر OK، يفتح وحدة النموذج أو التقرير عارضة قالب الشفرة لإجراء الحدث. يصف القسم التالي طريقة أسرع لعرض إجراء الحدث.

العمل في أسلوب عرض Module

يتضمن شريط أدوات أسلوب عرض Module أزرار لكل الأوامر.

تصف القائمة التالية الأزرار من اليسار لليمين وتتضمن اختصارات لوحة المفاتيح الخاصة بها.

- **View Microsoft Access:** يبدل بين Visual Basic Editor و.Microsoft Access (Alt+F11).
- **Insert:** انقر السهم لعرض الاختيارات ينشئ Module وحدة قياسية جديدة، يقوم Class Module بإنشاء وحدة مستقلة جديدة ويفتح Procedure حوار Insert Procedure ويدرج قالب شفرة للإجراء المحدد في الوحدة النشطة.
- **Save:** يحفظ التغييرات للحدث الحالي "Ctrl+S"
- **Cut:** يزيل النص الذي تم تحديده ويضع نسخة في Clipboard "Ctrl+X"
- **Copy:** يضع نسخة من النص الذي تم تحديده في Clipboard "Ctrl+C"

Paste: يلصق محتويات Clipboard عند نقطة الإدراج ويزيل الاختيار الحالي "Ctrl+V"

Find: يبحث عن النص الذي تم تحديده في الوحدة "Ctrl+F"

Undo: يتراجع عن لوحة المفاتيح القابلة للعكس الأكثر حداثة أو إجراء الفأرة. "Ctrl+Z"

Redo: يستعيد ما لم فعله بواسطة زر Undo

Go/Continue: يستكمل تنفيذ الشفرة بعد أن توقف التنفيذ إلا إذا منع التعليق الشفرة من التنفيذ. "F5"

Break: يوقف تنفيذ الإجراءات في الوحدة "Ctrl+Break"

Reset: ينهي تنفيذ الإجراءات في الوحدة ويبدأ من جديد كلا من المتغيرات الخاصة والعامة. "Alt+F5"

Design Mode: يبدل الإجراء داخل وخارج نوع التصميم.

Project Explorer: يعرض قائمة تنسيق شجرة لكل قواعد البيانات الحالية والمشاريع مع الوحدات بالإضافة إلى نماذج وتقارير تحتوي على وحدات "Ctrl+R"

Properties Window: يعرض أية خصائص قابلة للتغيير في الوحدة F4

Object Browser: يعرض مستعرض الكائن Object Browser "F2"

Tools: يعرض أدوات Visual Basic Editor

Help: يعرض إطار تعليمات Visual Basic Help "F1"

تحتوي القائمة في أسلوب عرض الوحدة Design على أوامر تساعدك على إنشاء وتحرير وتشغيل واستكشاف أخطاء الإجراءات وتصحيحها يسرد جدول ٧-٥ معظم أوامر القائمة الإضافية المحددة للوحدات التي ليست ممثلة بواسطة أزرار شرائط الأدوات الفرضية.

الجدول ٧-٥: أوامر القائمة التي تم تحديدها واختصارات لوحة المفاتيح

الأمر والاختصار	الوصف
Edit ⇦ Clear (Delete)	يزيل النص المحدد من إطار Module دون وضع نسخة في Clipboard.

الجدول ٧–٥: أوامر القائمة التي تم تحديدها واختصارات لوحة المفاتيح

الأمر والاختصار	الوصف
Select All ⇦ Edit (Ctrl+A)	يحدد كل النص في إطار window
Find (Ctrl+F) ⇦ Edit	يعرض حوار Find لتصميم بحث ويبحث عن التعبير المحدد في الإجراء الحالي أو الوحدة أو كل الوحدات كما تم تعيينها في حوار Find.
Find Next ⇦ Edit (F3)	يبحث عن التواجد التالي لسلسلة البحث في الإجراء الحالي أو الوحدة أو كل الوحدات كما تم تحديدها في حوار Find.
Replace ⇦ Edit (Ctrl+H)	يعرض حوار Replace لتصميم بحث ويحدد تعبيرات البحث والاستبدال.
Indent (Tab) ⇦ Edit	يعد السطور المحددة للشفرة أو سطر الشفرة عند نقطة الإدراج بأربعة مسافات" أو عدد المسافات المعدة كعرض Tab Width في حوار Options".
Outdent ⇦ Edit (Shift+Tab)	يزيح السطور المحددة للشفرة أو سطر الشفرة في نقطة الإدراج لليسار بأربعة مسافات" أو عدد المسافات المعدة مثل Tab Width في حوار Options".
List ⇦ Edit Properties/Methods (Ctrl+J)	يسرد خصائص وطرق لعنصر الشفرة أو العبارة التي تحتوي على نقطة إدراج.
List ⇦ Edit Constants (Ctrl+Shift+J)	يعرض ثوابت للعبارة التي تحتوي على نقطة إدراج.
Quick Info ⇦ Edit (Ctrl+I)	يعرض معلومات بناء الجملة للمتغير أو الثابت أو الإجراء الذي يحتوي على نقطة إدراج.
Parameter ⇦ Edit Info (Ctrl+Shift+I)	يعرض كل المعلمات للعبارة التي تحتوي على نقطة إدراج.

الجدول ٧-٥: أوامر القائمة التي تم تحديدها واختصارات لوحة المفاتيح

الأمر والاختصار	الوصف
Complete ⇦ Edit Word (Ctrl+Alt+A)	يستكمل طبع الخاصية المضمنة أو الطريقة أو جزء الكلمة الثابت عند نقطة الإدراج عارضاً قائمة بالاختيارات عندما تبدأ أكثر من كلمة بنفس الحروف.
Bookmarks ⇦ Edit	يعرض قائمة منطلقة بالأوامر الخاصة بالإعداد والإزالة والتنقل بين الإرشادات المرجعية.
Definition ⇦ View (Shift+F2)	يعرض شفرة إجراء لاسم الإجراء في نقطة الإدراج.
Last Position ⇦ View (Ctrl+Shift+F2)	يعود لموقع السطر عندما كنت تعرض في الإجراء السابق.
Compile ⇦ Debug (database)	يجمع كل الإجراءات في الوحدات المفتوحة.
Add Watch ⇦ Debug	يعرض حوار لتحديد تعبير ليتم إضافته للوح Watch الخاص Visual Basic Editor
Edit Watch ⇦ Debug	يعرض حوار Edit Watch لتحرير تعبير في لوح Watch إلى Visual Basic Editor.
Quick ⇦ Debug Watch (Shift+F9)	يسمح لك برؤية قيمة المتغير بينما يكون الإجراء في وضع التوقف.
Toggle ⇦ Debug Breakpoint (F9)	يبدل من نقطة الفصل من والي السطر الحالي.
Clear All ⇦ Debug Breakpoint (Ctrl+Shift+F9)	يزيل كل النقاط الفاصلة في كل الإجراءات في كل الوحدات في قاعدة البيانات الحالية.
Set Next ⇦ Debug Statement (Ctrl+F9)	يعين التعبير التالي في نفس الإجراء الذي سيتم تنفيذه موجود فقط في وضع التوقف.
Show Next ⇦ Debug Statement	يعرض العبارة التالية القابلة للتنفيذ

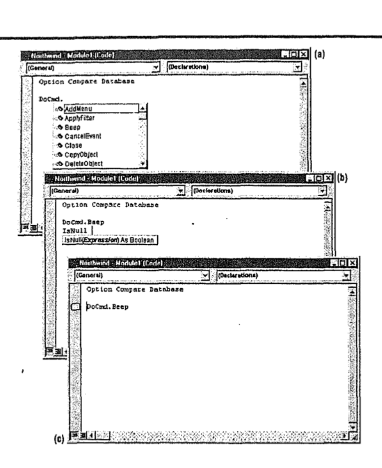
الجدول ٧-٥: أوامر القائمة التي تم تحديدها واختصارات لوحة المفاتيح

الأمر والاختصار	الوصف
Tools ⇦ References	يعرض حوار References لإضافة المراجع بمكتبات كائن أخر ولقواعد بيانات أخرى بحيث تستطيع استخدام إجراءات الخاصة بها.
Tools ⇦ Options	يفتح حوار Options
Window ⇦ Split	يبدل بين إطار Module المقسم في لوحين استعادة الإطار للوح واحد.

تتيح لك الأوامر المفيدة في قائمة Edit بعرض قوائم للخصائص والطرق والثوابت والمعلمات أو معلومات بناء الجملة للشفرة التي تم تحديدها في إطار Module. على سبيل المثال، إذا طبعت. DoCmd وحددت Edit ⇦ List Properties/Methods يعرض VBA مربع قائمة صغير في إطار Module بقائمة بطرق كائن DoCmd انقر مرتين عنصراً محدداً لإدراج العنصر في نقطة الإدراج انظر شكل "٧-٥ أ". إذا طبعت IsNull واخترت أمر Quick Info، يعرض VBA مربع نص صغير ببناء جملة للوظيفة المضمنة انظر شكل "٧-٥ ب".

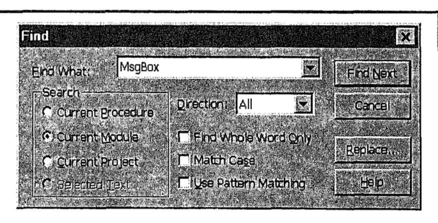
تتاح لك أيضاً إمكانية حفظ مكانك في إجراء ما بتعيين إشارة مرجعية لإعداد إشارة مرجعية. ضع نقطة الإدراج في سطر المسطرة الذي تريد تمييزه، اختر Edit ⇦ Bookmarks ثم اختر Toggle Bookmark من قائمة الانطلاق. يعرض VBA مستطيلاً أزرق في شريط مؤشر الهامش يسار سطر الشفرة انظر شكل "٧-١٥ ج". بالتبادل، يمكنك النظر يميناً جانب السطر الخاص بالشفرة ونقر Toggle ⇦ Bookmark. يمكنك تعيين ما تشاء من الإشارات المرجعية في وحدة واستعراضها باستخدام أمر Next Bookmark وأمر Previous Bookmark في قائمة Bookmark المنطلقة. يمكنك إزالة كل الإشارات المرجعية في الوحدة النشطة عن طريق اختيار أمر Clear All Bookmarks في القائمة المنطلقة.

تشبه كتابة شفرة VBA كتابة النص في الواقع، يمكنك إنشاء إجراءات في أي محرر نص مثل Notepad ثم إلصاقها في وحدات VBA. أدوات تحرير النص المعتادة متوفرة في أسلوب عرض Module. يمكنك قص ونسخ وإلصاق تحديدات النص باستخدام أوامر القائمة أو اختصارات لوحة المفاتيح العادية Ctrl+X وCtrl+C وCtrl+V "أو مسح تحديد دون وضع نسخة في Clipboard باختيار Edit ⇦ Clear" يمكنك أيضاً استخدام ميزة السحب والإسقاط بتحريك النص المحدد لموقع أخر.



الشكل ٧-١٥ استخدام أوامر القائمة لعرض قائمة بالخصائص والطرق "أ" أعرض بناء جملة متغير أو ثابت أو طريقة "ب" عين إشارة مرجعية للعودة سريعاً لسطر شفرة مميز "ج".

يمكنك استخدام أوامر Find وReplace العادية في قائمة Edit للبحث عن تعبير نص محدد أو للبحث عن تعبير نص محدد واستبداله بتعبير أخر. يمكنك البحث عن السلسلة المحددة في الإجراء الحالي أو في الوحدة الحالية أو في كل وحدات قاعدة المعلومات الحالية انظر شكل ٧-١٦.



الشكل ٧-١٦ استخدم حوار Find للبحث في الإجراء الحالي أو الوحدة الحالية أو كل الوحدات في قاعدة البيانات الحالية.

إعداد خيارات Visual Basic Editor

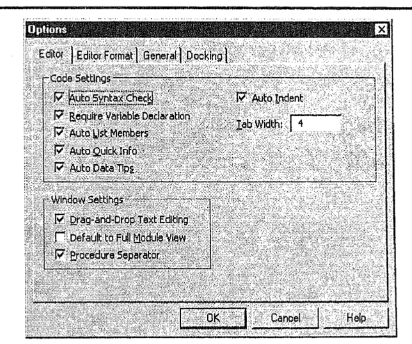
يمكنك إعداد خيارات متعددة لتلائم تفضيلاتك للعمل مع Visual Basic Editor. لتشغيل هذه الخيارات، افتح Visual Basic Editor بضغط Alt+F11 وتحديد Tools ⇦ Options. انقر جدولة Editor لرؤية الجدول الموضح في شكل ٧-١٧.

ملاحظة

في اكسس ٩٧، كان لحوار Options جدولة Modules احتوت على إعدادات تؤثر في إطار Module وإطار Immediate وObject Browser وإطار Project Browser وإطار Properties والخيارات التي كانت سابقاً في حوار Options موجودة الآن في حوار Visual Basic Editor Options.

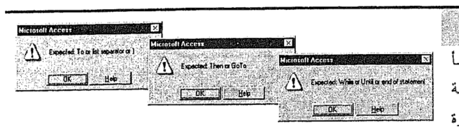
الشكل ٧-١٧

جدولة Editor Visual Basic Editor Options.



تحتوي جدولة Editor لحوار Options على الخيارات التالية:

Auto Syntax Check: يدقق لأخطاء بناء الجملة بينما تقوم أنت بطباعة الشفرة يوضح شكل ٧-١٨ رسائل تدقيق أخطاء بناء الجملة التقليدية. بتشغيل Auto Syntax Check، يمكن أن يعيد VBA تنسيق العبارة بإضافة أو إزالة المسافات ويمكنه أن يغير جعل حروف الكلمات الأساسية كبيرة والمتغيرات لتلائم هذا التكبير في عبارة إعلان المتغير ويمكنه أن يصحح أخطاء بناء الجملة الصغيرة مثل إضافة علامة تنصيص مزدوجة ناقصة.



الشكل ٧-١٨
رسائل تدقيق خطأ بناء الجملة التقليدية. عبارة "أ" تنقصها أقواس أخطأت في تهجي كلمة أساسية في تركيب "ب" حذفت جزء مطلوب من التركيب "ج".

Auto Indent: يضيف مسافة بادئة تلقائية لسطر متتابع لشفرة في نفس عدد وقفات الجدولة كالسطر الحالي.

Require Variable Declaration: يتطلب إعلان كل المتغيرات بوضوح بتدقيق هذا الخيار، يتم تضمين عبارة Option Explicit في عبارة Declarations لكل الوحدات الجديدة.

Auto List Members: يعرض قائمة بالخيارات الصالحة لعنصر الشفرة الذي طبعته عندما تشير لكائن أو لمتغير كائن. اضغط Ctrl+Enter أو Tab لإدخال العنصر المحدد اضغط Esc لإغلاق القائمة.

Auto Quick Info: يعرض بناء الجملة للإجراء أو الطريقة عندما تطبع الاسم متبوعاً بمسافة أو فترة أو أقواس فتح.

Auto Data Tips: تعرض القيمة الحالية للمتغير الذي تم تحديده عندما تكون وضع توقف وتوقف مؤشر الفأرة فوق المتغير.

Tab Width: يعين عدد المسافات بين وقفات الجدولة في إطار Module. الفرض هو ٤. يمكنك تغيير التعيين لأي رقم بين أو ٣٢.

Drag-and-Drop Text Editing: يتيح لك تحديد النص وسحبه لمكان أخر في إطار Module أو Immediate.

Default to Full Module View: يعرض قسم Declarations وكل الإجراءات في نفس اللوح مفصولة بأسطر أفقية. "أزل ضغط هذا الخيار لإزالة أسطر فصل الإجراء."

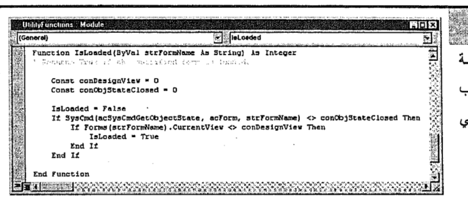
Procedure Separator: ويفصل قسم Declarations وكل إجراء بخطوط أفقية. "قم بإلغاء اختيار هذا الخيار لحذف خطوط فاصل الإجراء".

اتباع نمط برمجة جيد

في أثناء كتابتك للشفرة، اعتبر النصائح التالية لجعل شفرتك أسهل في القراءة والفهم.

Indent: بطبيعة الحال، للإجراء أقسام متعددة. ربما يكون هناك قسم لشفرة التعامل مع الأخطاء أو مجموعات أسطر شفرة للعمليات المكررة أو مجموعات أخرى للعبارات التي ستنفذ تحت بعض الظروف ولكن ليس تحت أخرى. أسطر المسافات البادئة وسيلة مفيدة للتعرف على أقسام الشفرة يوضح شكل ٧-١٩ إجراء وظيفة IsLoaded الذي يحدد إذا كان نموذج محدد قد تم تحميله. حتى قبل دراسة كيفية كتابة الشفرة التي تصنع القرارات يمكنك ملاحظة بناء الشفرة بسهولة بفضل مستويات المسافات البادئة.

الشكل ٧-١٩ المسافات البادئة تجعل تركيب الشفرة أسهل في الفهم.



```
Function IsLoaded(ByVal strFormName As String) As Integer

    Const conDesignView = 0
    Const conObjStateClosed = 0

    IsLoaded = False
    If SysCmd(acSysCmdGetObjectState, acForm, strFormName) <> conObjStateClosed Then
        If Forms(strFormName).CurrentView <> conDesignView Then
            IsLoaded = True
        End If
    End If

End Function
```

Use comments: في أثناء إنشائك للشفرة، يمكن أن يكون واضحاً الغرض والمنطق من عبارة الإجراء بعد ذلك من الممكن أن يكون واضحاً بالنسبة لآخرين. يجب أن تضمن تعليقات في بداية بواسطة إجراء وظيفة. بينما من غير الضروري التعليق على كل سطر من الشفرة، يجب على الأقل أن تضمن تعليقاً لشرح منطق كل مجموعة من العبارات تسبق أسطر التعليقات بفاصلة علوية واحدة.

Use naming conventions: يناقش الفصل التالي أهمية اصطلاحات التسمية لتسمية الكائنات التي تنشئها في التطبيق الخاص بك بوجود اصطلاح تسمية تم اختياره بصورة صحيحة ومترابطة للإجراءات والكائنات والمتغيرات، يصبح شفرة VBA موثقة لنفسها وأسهل في الفهم.

Use separate lines for each statement: بالرغم من إمكانية سلسلة عدة عبارات مفصولة بفاصلة منقوطة (:) في سطر واحد فعل ذلك سيجعل الشفرة صعبة في القراءة.

Use the line-continuation character: لا يحتوي إطار Module على ميزة إحاطة بالكلمة. إذا أدخلت عبارة طويلة يمكنك "إحاطة" العبارة بنفسك باستخدام line-continuation character. يتكون حرف استكمال السطر من مسافة متبوعة بتسطير أسفل السطر (_). إذا لم تفصل السطر بحرف استكمال السطر، ستحتاج لاستخدام كرة التحرير الأفقية لعرض أجزاء من العبارة. لا يمكنك استخدام حرف استكمال السطر لإحاطة تعبير سلسلة أخر، بدلاً من ذلك، يمكنك تقسيم السلسلة لأجزاء صغيرة وسلسلة الأجزاء انظر الفصل ١٣ للأمثلة.

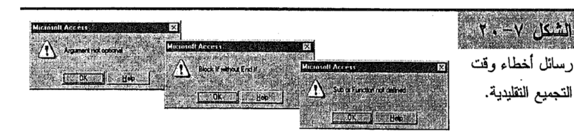
Declare variables and constants at the beginning of a procedure: بتجميع كل عبارات الإعلان في بداية إجراء يمكنك رؤية الثوابت والمتغيرات بسرعة وتجنب الحاجة للتجول في شفرة الإجراء للعثور عليهم.

استخدام مجمع Access VBA

تدعى العبارات التي تدخلها في وحدة مصدر شفرة source code لا يقوم أكسس بتشغيل العبارات في أثناء إدخالات لها. قبل أن تشغل إجراء، يجب أولاً أن يقوم أكسس بتجميعه. التجميع هو عملية تحويل مصدر الشفرة القابل للقراءة والذي أدخلته في التنسيق المدعو compiled state الذي يستطيع الحاسب تشغيله يهيئ التجميع الشفرة الخاصة بك للتنفيذ ولكنه لا ينفذها. في أثناء التجميع، يعرض أكسس الإجراء ككل ويدقق للأخطاء التي تتطلب أكثر من عبارة. تدعى الأخطاء التي يحددها أكسس في أثناء التجميع أخطاء وقت التجميع عندما يكتشف أكسس أخطاء وقت تشغيل يتم عرض رسالة خطأ. يوضح شكل ٧-٢٠ أخطاء وقت تشغيل تقليدية لا يشغل أكسس الإجراء حتى تحذف مصدر الخطأ.

الشكل ٧-٢٠

رسائل أخطاء وقت التجميع التقليدية.



حفظ قاعدة معلومات دون مصدر شفرة

يمكنك أيضاً حفظ إصدار قاعدة معلومات بشفرة VBA كملف mde file الذي يتضمن الإصدارات المجمعة وبها مصدر شفرة تم إزالته له الميزات التالية:

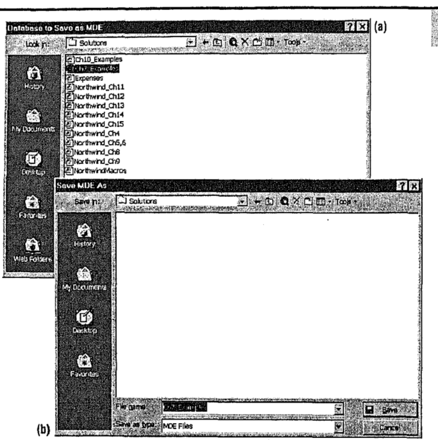
- ♦ لا يمكن تغيير النماذج والتقارير والوحدات.
- ♦ لا يمكن رؤية الوحدات. يتم تأمين الشفرة الخاصة بك تماماً ولا يستطيع آخرون قراءتها أو استخدامها.
- ♦ ملف mde. أصغر من ملف قاعدة البيانات الأصلي.
- ♦ استخدام الذاكرة يتم جعله مثالياً ويمكن أن يتحسن الأداء.
- ♦ لا يمكن استيراد أو تصدير النماذج والتقارير والوحدات مع ذلك يمكن استيراد أو تصدير الجداول والاستعلامات والماكروات لقواعد بيانات ليس لديها mde.
- ♦ لا يمكنك تغيير المراجع لمكتبات الكائن أو لقواعد البيانات.
- ♦ لا يمكنك تغيير اسم مشروع VBA لقاعدة البيانات.

لإنشاء ملف mde اتبع الخطوات التالية:

١- أغلق قاعدة البيانات إذا كنت في بيئة متعددة المستخدمين يجب إغلاق كل نسخ قاعدة البيانات.

٢- اختر Tools ⇦ Database Utilities ثم أمر Make MDE File في القائمة المنطلقة حدد قاعدة البيانات في Database إلى حوار Save as MDE انظر "شكل ٧-٢١ أ" ثم انقر زر Make MDE.

٣- حدد اسم الملف الجديد وحدد المجلد لحفظ الملف في حوار Save MDE As انظر شكل "٧-٢١ ب" ثم انقر زر Save.



الشكل ٧-٢١

إنشاء ملف mde بوجود مصدر شفرة VBA وقد تم إزالته.

تحذير يجب عليك دائماً حفظ الإصدار الأصلي من قاعدة المعلومات التي استخدمتها لإنشاء ملف mde. إذا احتجت لتغيير النماذج أو التقارير أو الوحدات، يجب أن تأخذ التغييرات لقاعدة البيانات الأصلية ثم تقوم بإنشاء ملف mde جديد بالإضافة إلى ذلك لن تتمكن من تشغيل أو تحويل ملف mde في إصدارات تالية من أكسس، لكن ستكون قادراً على تحويل ملف قاعدة البيانات الأصلي.

هناك بعض القيود بالنسبة لحفظ قاعدة بيانات كملف mde على سبيل المثال، إذا أشارت قاعدة بيانات لقواعد بيانات أخرى أو إضافات بمعنى أنه إذا قمت بتعيين مرجع لقاعدة بيانات أخرى أو إضافة في حوار References، يجب أن تحفظ كل قواعد البيانات الأخرى والإضافات كملفات mde. لمزيد من المعلومات على القيود، أبحث عن "mde files" في التعليمات الفورية.

طرق لتشغيل الإجراءات

يناقش هذا القسم استدعاء الإجراءات وتستعمل كلمة استدعاء وكلمة تشغيل بالتبادل لإعطاء نفس المعنى. لأن إجراء وظيفة يستطيع إرجاع قيمة ولا يستطيع الإجراء الفرعي فعل ذلك، فلديك مرونة أكثر في استدعاء إجراءات الوظيفة، على سبيل المثال، يمكنك استدعاء إجراء وظيفة لكن ليس إجراء فرعياً في تعبير ما.

تشغيل إجراءات الوظيفة

فيما يلي قائمة بالطرق التي تستطيع عن طريقها استدعاء إجراء وظيفة:

- استدع في تعبير
- شغل في إطار Immediate
- استدع من إجراء في نفس الوحدة
- استدع من إجراء في وحدة أخرى
- أطلق عن طريق حدث
- استدع من ماكرو
- استدع من تطبيق أخر

عندما تستدع إجراء وظيفة في تعبير يستخدم VBA قيمة إرجاع الإجراء في تقييم التعبير في كل الطرق الأخرى لاستدعاء وظيفة، يقوم VBA ينبذ قيمة الإرجاع.

استدعاء إجراء وظيفة في تعبير ما

يمكنك استدعاء إجراء وظيفة في تعبير بنفس الطريقة التي تستدعي بها وظيفة مضمنة. يمكنك أن تضمن إجراء وظيفة في تعبيرات لحقول محسوبة أو خلايا معايير في الاستعلامات وفي تعبيرات لتحكمات محسوبة في النماذج والتقارير وفي أحوال ماكرو ووسائط إجراء وفي إعدادات خاصية وفي تعبيرات مستخدمة في عبارات VBA وعبارات SQL. على سبيل المثال، وظيفة Concatenate الموجودة أسفل مفيدة لسلسلة مقطعي اسم الشخص للمقطع الأخير وتنسيق الاسم الأول:

```
Public Function Concatenate (A,B)
   Concatenate = A & ", " & B
   MsgBox "This is the Concatenate procedure in bas2"
End Function
```

الوظيفة لها الوسيطان A و B عندما تمرر قيماً للوسائط، تقوم الوظيفة بمسلسلة القيم بالسلسلة "و" بينهما وترجع السلسلة. الرؤية الوظيفة وهي تعمل، اتبع هذه الخطوات:

١- قم بإنشاء وحدة قياسية جديدة تدعى bas2 in Ch7_Examples اختر Insert ⇦ Procedure وقم بإنشاء إجراء جديد يدعى Concatenate بنوع Function ومجال Public.

٢- أدخل السطرين الخاصين بالشفرة لإجراء وظيفة Concatenate أعلى.

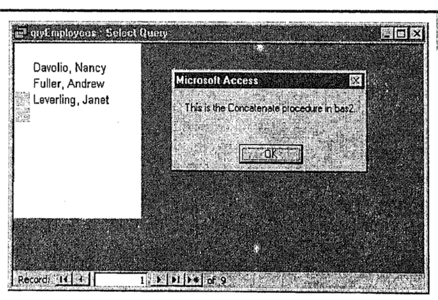
٣- استورد جدول Employees من قاعدة بيانات Northwind قم بإنشاء استعلام جديد يدعى qryEmployees معتمداً على جدول Employees وادخل التعبير التالي في خلية Field الأولى:

FullName: Concatenate(LastName, FirstName)

٤- قم بتشغيل الاستعلام.

عندما تشغل الاستعلام، ينفذ VBA التتابع التالي لكل صف في نتيجة الاستعلام. أولاً يقوم بحساب قيمة الإرجاع للوظيفة ويحمل القيمة في الذاكرة، ثم يعرض الرسالة ثم ينشئ ويعرض الحقل المحسوب في ورقة بيانات الاستعلام انظر شكل ٧-٢٢.

ملاحظة تقوم بتضمين وظيفة MsgBox في معظم الإجراءات في الفصل كأداة تعلم أو كأداة لاستكشاف الأخطاء وإصلاحها لملاحظة متى يعمل الإجراء. يمكن أن تعلق أو تحذف العبارات بوظيفة Msg Box عندما تستخدم الإجراءات في تطبيق.



الشكل ٧-٢٢ تعبير خلية Field في الاستعلام يستدعي وظيفة Concatenate لكل صف في نتيجة الاستعلام.

تشغيل إجراء وظيفة في إطار Immediate

إطار Immediate أداة مفيدة لتشغيل الإجراءات ولأنـــه أداة تصحيـــح لإطـــار Immediate مميزات رؤية أوضح فهو يرى ويمكنه تشغيل كل من إجرائي public وprivate إذا اســـتخدمت مرجعاً مؤهل تماماً. يعتمد بناء الجملة الذي تستخدمه لتشغيل إجراء وظيفة على مــــا إذا كـــان مخزناً في وحدة قياسية أو وحدة نموذج أو تقرير وما إذا كنت تريد استخدام قيمة الإرجاع.

تشغيل إجراءات وظيفة الوحدات القياسية

إذا كانت الوظيفة مخزنة في وحدة قياسية، يمكنك طبع قيمة الإرجاع بطبع ? functionname (argumentlist) وضغط Enter إذا أردت استخدام قيمة الإرجاع في عبارة أخـــرى، يمكنـــك إنشاء متغير في إطار Immediate لجعل قيمة إرجاع الوظيفة إذا أرجعت الوظيفة كائناً، استخدم بناء الجملة:

```
var = functionname (argumentlist)
If the function returns an object, use the syntax:
Set objvar = functionname (argumentlist)
```

على سبيل المثال، افتح إطار Immediate بضغط Ctrl+G وأطبع العبارات التالية:

- أطبع ("Concatenate("Johnson","Diane ? واضغط Enter تعود الوظيفة ويتـــم عرض الرسالة طبع النتيجة في إطار Immediate.
- أطبع ("var = Concatenate("Johnson","Diane واضغط Enter يتـــم عـــرض الرسالة وتخزين القيمة المرجعة في المتغير.
- أطبع var? واضغط Enter يتم طبع النتائج المخزنة في إطار Immediate.

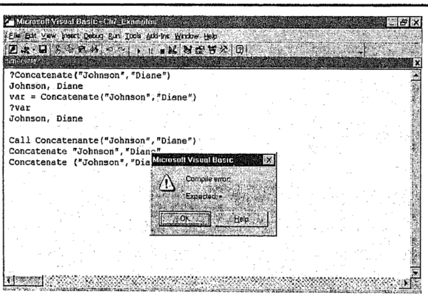
إذا كنت لا تريد قيمة إرجاع وظيفة أن تكون مخزنة في وحدة قياسية، يمكنك تشغيل الوظيفـــة في إطار Immediate باستخدام واحدة مما يلي:

```
Call functionname (argumentlist)
functionname argumentlist
```

إذا استخدمت الكلمة الأساسية Call لاستدعاء إجراء وظيفة يتطلب وسائط، يجب أن تضمـــن قائمة الوسيطة في أقواس وإذا لم تتطلب الوظيفة وسائط، يمكنك أن تضمن أو تلغي الأقـــواس. إذا حذفت الكلمة الأساسية Call، يجب أن تلغي الأقواس التي تحيط بقائمة الوسيطة أيضـــاً. عندمـــا تستخدم أياً من هذه العبارات، يلقي VBA بقيمة إرجاع الوظيفة. على سبيل المثال، أطبع العبارات التالية في إطار Immediate.

- أطبع Call Concatenate ("Johnson", "Diane") واضغط Enter يشغل VBA الوظيفة ويعرض الرسالة ويلغي بقيمة إرجاع الوظيفة تتحرك نقطة الإدراج للسطر التالي دون طبع أي شيء.
- أطبع Concatenate "Johnson", "Diane" واضغط Enter وستحصل على نفس النتيجة.
- أطبع Concatenate ("Johnson", "Diane") واضغط Enter يولد VBA خطأ ويعرض رسالة خطأ انظر شكل ٧-٢٣.

الشكل ٧-٢٣ رسالة الخطأ عندما تحذف الكلمة الأساسية Call لكن تضمن الوسائط في أقواس.



يمكنك تشغيل كلاً من إجراءات الوظيفة العامة والخاصة في إطار Immediate. لتشغيل وظيفة خاصة في إطار Immediate يجب أن تستخدم المعرف الكامل. اتبع الخطوات التالية لرؤية كيفية عمل هذا:

١- غير Public إلى Private ككلمة أساسية في عبارة إعلان وظيفة Concatenate في وحدة bas2.

٢- أطبع ?bas2.Concatenate ("Johnson", "Diane") واضغط Enter يشغل VBA إجراء Concatenate ويعرض الرسالة ويعرض القيمة المرجعة.

٣- أطبع ?Concatenate ("Johnson", "Diane") واضغط Enter يولد VBA خطأ وقت تشغيل عندما تحاول تشغيل وظيفة خاصة دون تضمين اسم الوحدة في المرجع.

٤- غير Private مرة أخرى إلى Public ككلمة أساسية في عبارة إعلان وظيفة Concatenate.

تشغيل إجراءات وظيفة وحدة النموذج أو التقرير إذا كان إجراء الوظيفة مخزناً في وحدة نموذج أو تقرير يجب أن تستخدم المعرف الكامل للوظيفة في هذه التعبيرات على سبيل المثال، إذا أرجعت وظيفة في وحدة نموذج قيمة ما، يمكنك طبع القيمة في إطار Immediate باستخدام بناء الجملة:

```
? Form_ formname.functionname (argumentlist)
```

يمكنك تشغيل الوظيفة وبند قيمة إرجاعها باستخدام:

```
Call Form_ formname.functionname (argumentlist)
```

لا يحتاج النموذج أو التقرير أن يكون مفتوحاً لتشغيل إجراء في وحدته. في إطار Immediate اتبع هذه الخطوات:

١- أطبع (3)Form_frm1.Function1 ? واضغط Enter يشغل VBA الوظيفة ويعرض الرسالة ويطبع ٤ "تضيف Function1 واحد للوسيطة وترجع المجموع"، تأكد من إغلاق نموذج frm1 عندما تستدعي الوظيفة.

٢- أطبع (3)Function1? واضغط Enter. يتم تنفيذ إجراء Function1 لوحدة bas1، مرجعاً قيمة تساوي ٥.

٣- افتح وحدة نموذج bas1 وغير Public إلى Private يكون الإجراء الآن غير متوفر لأية وحدة سوى وحدة bas1. أطبع (3)Function1? في إطار Immediate. عندما تحاول تشغيل وظيفة في نموذج أو تقرير في إطار Immediate دون تضمين المرجع للوحدة يحدث خطأ وقت تشغيل "Sub or Function not defined" غير Function1 لوحدة عامة.

استدعاء من إجراء في نفس الوحدة

عندما تستدعي إجراء وظيفة من وظيفة أو إجراء فرعي، يعتمد بناء الجملة الذب تستخدمه على ما إذا كنت تريد استخدام قيمة إرجاع الوظيفة أم لا.

إذا أردت استخدام قيمة الإرجاع، يجب أن تستخدم إجراء الوظيفة في تعبير في عبارة VBA. على سبيل المثال، اتبع هذه الخطوات:

١- قم بإنشاء إجراء وظيفة جديد. يدعى GetReturnValue في وحدة bas2 كما هو موضع فيما يلي تستدعي وظيفة GetReturnValue وظيفة Concatenate كوسيطة لوظيفة MsgBox ويعرض القيمة المرجعية في مربع رسالة:

```
Public Function GetReturnValue()
   MsgBox Concatenate ("Johnson", "Diane")
End Sub
```

٢- شغل وظيفة GetReturnValue في إطار Immediate بطبع Call GetReturnValue() واضغط Enter. يمكنك أيضاً استدعاء الوظيفة بطبع GetReturnValue واضغط Enter. ولأن وظيفة GetReturnValue ليس لها وسائط، يمكنك أيضاً استدعاء الوظيفة بنقر Call GetReturnValue واضغط Enter. لاحظ أن وظيفة GetReturnValue ليس لها قيمة إرجاع.

إذا لم يكن محتاجاً لقيمة الإرجاع، يمكنك تشغيل إجراء الوظيفة من وظيفة أخرى أو إجراء فرعي باستخدام أي من العبارتين التاليتين:

```
Call functionname (argumentlist)
functionname argumentlist
```

إذا كنت تستطيع الكلمة الأساسية Call لاستدعاء إجراء وظيفة وسائط، يجب أن تضمن قائمة الوسائط في أقواس. إذا لم تتطلب الوظيفة وسائط، يمكن أن تضمن أو تحذف الأقواس. إذا لم تستخدم الكلمة الأساسية Call يجب أن تحذف الأقواس حول قائمة الوسيطة في كلا الحالتين، يتم بتجاهل إرجاع الوظيفة. على سبيل المثال، تشغل وظيفة DiscardReturn التالية وظيفة Concatenate متجاهلة قيمة الإرجاع لا تأخذ وظيفة DiscardReturn أية وسائط ولا ترجع قيمة ما.

```
Public Function DiscardReturnValue()
   Call Concatenate ("Johnson", "Diane")
End Sub
```

اتبع هذه الخطوات لاختبار وظيفة DiscardReturnValue:

١- أدخل وظيفة DiscardReturnValue في bas2. وضغط Enter. يشغل VBA وظيفة Concatenate ويعرض الرسالة ويتجاهل قيمة الإرجاع.

٢- عدل الإجراء بحذف كل من الكلمة الأساسية Call والأقواس كما يلي:

```
Public Function DiscardReturnValue()
   Concatenate "Johnson", "Diane"
End Sub
```

٣- شغل الوظيفة بطبع DiscardReturnValue وضغط Enter استدعاء من إجراء في وحدة أخرى.

٤- شغل الوظيفة بطبع DiscardReturnValue وضغط Enter.

استدعاء من إجراء في وحدة أخرى

يمكن أن تستدعي إجراء وظيفة عام وليس خاص من إجراء محفوظ في وحدة أخرى. إذا كنت تستدعي إجراء وظيفة عام في وحدة أخرى، يمكن أن تحتاج لتضمين الاسم في المرجع. على

سبيل المثال، إجراء الوظيفة RunFormFunction الموضح بالأسفل يشغل وظيفة Function1 العامة في وحدة النموذج لنموذج frm1.

```
Public Function RunPublicFunction()
   MsgBox Form_frm1.Function1(3)
End Function
```

اختبر الإجراءات العامة والخاصة كما يلي

١- أدخل إجراء RunPublicFunction في وحدة bas2.

٢- أطبع Call RunPublicFunction في إطار Immediate واضغط Enter. يشغل إجراء Function1 ويعرض مربعات الرسالة.

٣- أدخل إجراء RunPrivateFunction في وحدة bas2 كما يلي:

```
Public Function RunPrivateFunction()
   MsgBox bas1.Function1(3)
End Function
```

٤- أطبع Call RunPrivateFunction في إطار Immediate واضغط Enter. يولد VBA خطأ وقت تشغيل "Method or data member not found" لأنك لا تستطيع تشغيل إجراء خاص مخزن في وحدة أخرى يميز VBA اسم الإجراء ويعرض سهماً سهماً في شريط الهامش ويغير لون خلفية العبارة التي فشلت.

٥- انقر زر Reset في شريط الأدوات لإعادة تعيين الشفرة.

إطلاق إجراء بواسطة حدث

يمكنك استخدام حدث لإطلاق وظيفة لتشغيل وظيفة عندما يحدث حدث ما، استخدم بناء الجملة التالي في مربع خاصية الحدث:

```
= functionname(argumentlist)
```

الأقواس مطلوبة حتى إذا لم تحتوي الوظائف على وسائط يتم تجاهل قيمة إرجاع الوظيفة إن وجدت.

اتبع هذه الخطوات لإطلاق وظيفة عن طريق حدث:

١- حدد نموذج frm1 في إطار Database وانقر Code في شريط الأدوات. يفتح النموذج والوحدة الخاصة به.

٢- انقر زر Insert في شريط الأدوات واختر Procedure سمي الإجراء الجديد EventFunction واختر نوع Function ومجال Public.

٣- أدخل EventFunction التالي لا يأخذ هذا الإجراء أية وسائط ويرجع قيمة Boolean إلى True.

```
Public Function EventFunction()
    EventFunction = True
    MsgBox "This is the EventFunction in the Form_frm1 module"
End Function
```

٤- افتح ورقة خاصية النموذج، انقر مربع خاصية الحدث OnClick وأطبع = EventFunction() احفظ النموذج وبدل لأسلوب عرض Form.

٥- عندما تنتهي وظيفة Function1 من التشغيل، انقر مرتين في قسم تفاصيل النموذج. يشغل VBA إجراء EventFunction ويعرض مربع الرسالة الخاص به ويتجاهل قيمة إرجاع الوظيفة.

يستخدم بناء الجملة لإعدادات خاصية الحدث دون اسم وحدة مؤهل في واقع الأمر لا تستطيع استخدام المرجعي المؤهل كاملاً كإعداد خاصية حدث هذا يعني أنك تستطيع استخدام حدث لإطلاق وظيفة عامة مخزنة وحدة قياسية طالما كان اسم الوظيفة مزيداً ولا يتطلب المرجع المؤهل كاملاً لكنك لا تستطيع إطلاق وظيفة عامة مخزنة في نموذج أخر أو وحدة تقرير لأن استخدام وظيفة في نموذج أخر أو وحدة تقرير تتطلب المرجع المؤهل كاملاً جرب هذا المثال:

١- بدل لأسلوب عرض Design، انقر في مربع نص txtCalculate ثم انقر في خاصية OnClick سنستخدم حدث Click لمربع نص txtCalculate لإطلاق وظيفة Concatenate المخزنة في الوحدة القياسية bas2.

٢- أطبع ("Johnson", "Diane") Concatenate = في مربع الخاصية احفظ النموذج وبدل لأسلوب عرض Form.

٣- عندما تنتهي وظيفة Function1 من التشغيل، انقر مربع النص يشغل VBA وظيفة Concatenate ويعرض الرسالة الخاصة به ويتجاهل القيمة المرجعية.

استدعاء إجراء من ماكرو

لتشغيل إجراء وظيفة من ماكرو، استخدم إجراء ماكرو RunCode استخدم بناء الجملة التالي لوسيطة اسم وظيفة الإجراء.

```
functionname (argumentlist)
```

ضمن الوسائط في أقواس ولا تستخدم علامة تساوي (=) يتم تجاهل إرجاع الوظيفة. يتطلب بناء الجملة لوسيطة اسم الوظيفة. اسم وظيفة غير مؤهل هذا يعني أنك تستطيع استخدام إجراء RunCode لتشغيل وظيفة عامة مخزنة في الوحدة القياسية إذا لم تحتج لتأهيل اسمها بالإضافة

إلى ذلك، إذا أشتغل إجراء ماكرو متجاوباً مع حدث في نموذج أو تقرير، يبحث أكسس عن الوظيفة أولاً في وحدة النموذج أو التقرير قبل النظر في الوحدات القياسية لذلك في هذه الحالة يمكنك تشغيل وظيفة مخزنة في وحدة نموذج أو تقرير.

لمثال عن كيفية تشغيل وظيفة في وحدة قياسية، سنشغل وظيفة Concatenate:

١- افتح صفحة ماكرو جديدة تدعى mcrRunCode. حدد إجراء RunCode في خلية الإجراء الأولى وقم بتعيين وسيطة اسم الوظيفة إلى ,"Johnson") Concatenate ("Diane"). يوضح شكل ٧-٢٤ ورقة ماكرو.

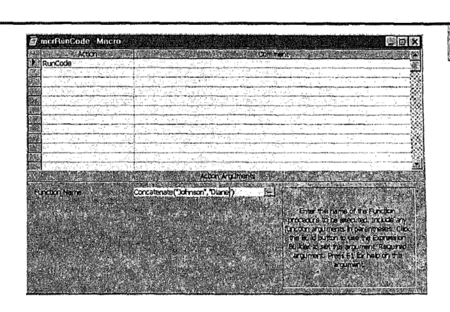
٢- احفظ ورقة الماكرو انقر زر Run في شريط أدوات Macro يشغل أكسس الماكرو. يشغل الماكرو وظيفة Concatenate التي تعرض الرسالة الخاصة بها وتتجاهل قيمة إرجاع الوظيفة ثم تنسحب.

استدعاء إجراءات تطبيق أخر

يمكنك Run لتشغيل وظيفة تطبيق أخر خلال Automation مستخدماً بناء الجملة:

```
Application.Run functionname[, argument1, argument2, ..., argumentN]
```

يتجاهل أكسس قيمة إرجاع الوظيفة.



الشكل ٧-٢٤

يمكنك تشغيل إجراء وظيفة من ماكرو.

تشغيل الإجراءات الفرعية

فيما يلي قائمة بالطرق التي يمكن عن طريقها استدعاء إجراء فرعي:

- أطلق بواسطة حدث

- ♦ تشغيل في إطار Immediate
- ♦ استدع من إجراء أخر
- ♦ استدع من تطبيق أخر

لأن الإجراء الفرعي لا يرجع قيمة، لا يمكنك استخدام إجراء فرعي في تعبير إذا أردت تشغيل إجراء فرعي كجزء من تعبير، يمكنك إنشاء وظيفة تقوم باستدعاء الروتين الفرعي ثم تستخدم الوظيفة في التعبير.

لا يمكنك أيضاً استدعاء إجراء فرعي مباشرة من ماكرو مع ذلك يمكنك إنشاء إجراء وظيفة يستدعي الإجراء الفرعي ويشغل الوظيفة باستخدام إجراء ماكرو RunCode.

إطلاق إجراء فرعي بواسطة حدث

عندما تقوم بإنشاء إجراء حدث بنقر زر Build يمين مربع خاصية الحدث وتدخل شفرة بين سطور قالب الشفرة التي يوفرها أكسس، يتم تعيين الإجراء الفرعي تلقائياً للحدث ويشتغل عندما يتم التعرف على الحدث. يقوم أكسس تلقائياً بتسمية الإجراء باستخدام بناء الجملة Form_eventname أو Report_eventname لحدث تم التعرف عليه بواسطة نموذج أو تقرير وبناء الجملة objectname_eventname لحدث تم التعرف عليه بواسطة كائن. يتضمن أكسس أيضاً قائمة الوسيطة المعرفة سابقاً لإجراء الحدث. بعد تسمية الإجراء يعين أكسس خاصية الحدث "Event Procedure". إذا أردت إنشاء إجراء حدث من لا شيء، يجب عليك اتباع نفس القواعد:

- ♦ يجب أن تستخدم نفس اصطلاح التسمية ونفس قائمة الوسيطة
- ♦ يجب أن تعين خاصية الحدث في "Event Procedure"

كمثال، ستقوم بإنشاء إجراء حدث من لا شيء لإغلاق نموذج:

١- افتح نموذج frm1 في أسلوب عرض Design، ضع زر أمر في النموذج وعين خاصية Name في cmdClose وخاصية Caption في Close.

٢- انقر زر Code في شريط الأدوات وضع نقطة الإدراج في السطر الذي يلي العبارة الأخيرة في الوحدة.

٣- أدخل الإجراء التالي في نقطة الإدراج:

```
Private Sub cmdclose_Click()
    DoCmd.Close
    MsgBox "This is cmdclose_Click in module Form_frm1"
End Sub
```

٤- عندما تدخل السطر الأول، يدرج VBA تلقائياً فاصل إجراء ويستكمل قالب الشفرة. يلائم أكسس اسم إجراء الحدث مع الزر الجديد ويعين تلقائياً الإجراء لحدث الزر Click انقر في ورقة الخاصية لزر الأمر لرؤية التعيين.

٥- احفظ النموذج وبدل لأسلوب عرض Form. يشتغل إجراء Function1.

٦- اقر زر Close. يغلق أكسس النموذج ثم يعرض الرسالة لإجراء حدث cmdClose_Click.

إذا غير اسم تحكم بعد تعيين إجراء حدث له. لا يكون VBA قادراً على مطابقة إجراء الحدث للتحكم. يحرك VBA إجراء الحدث لقسم General في الوحدة. يجب أن تغير اسم إجراء الحدث حتى يطابق اسم التحكم الجديد. حدد "General" من مربع التحرير والسرد Object، حدد إجراء الحدث من مربع التحرير والسرد Procedure غير اسم إجراء الحدث واحفظ النموذج.

ملاحظة عندما تنسخ وتلصق تحكماً مع إجراء مع إجراءات الحدث لنموذج أخر، لا يتم إجراء الحدث لوحدة النموذج للنموذج الجديد إذا أردت أن تكون قادراً على نسخ ولصق الإجراءات مع التحكمات استخدم إجراءات وظيفة تعامل مع الحدث بدلاً من إجراءات الحدث وخزن الوظائف في وحدة قياسية.

تشغيل إجراء فرعي في إطار Immediate

تشغل إجراءً فرعياً في إطار Immediate باستخدام بناء الجملة نفسه الذي تستخدمه لتشغيل وظيفة بتجاهل قيمة الإرجاع. يعتمد بناء الجملة الذي تستخدمه على ما إذا كان الإجراء الفرعي المخزن في وحدة قياسية أو وحدة نموذج أو من بنائي الجملة التاليين:

```
Call subroutinename (argumentlist)
subroutinename argumentlist
```

إذا تم تخزين الإجراء الفرعي في وحدة نموذج أو تقرير، يجب أن تستخدم المرجع المؤهل تماماً للوظيفة الموجودة في هذه التعبيرات. على سبيل المثال:

```
Form_ formname.subroutinename argumentlist
```

يمكنك تشغيل كلاً من الإجراءات الفرعية العامة والخاصة في إطار Immediate. يجب أن يفتح النموذج أو التقرير حتى يشغل أجزاء فرعياً في الوحدة الخاصة به. على العكس، لا يحتاج النموذج أو التقرير للفتح حتى يشغل في إطار Immediate إجراء وظيفة مخزن في الوحدة الخاصة به. على سبيل المثال. لاستدعاء إجراء الحدث من زر cmdButton1 في نموذج

frm1، افتح نموذج frm1 في أسلوب عرض Design أو Form أطبع Call Form_frm1.cmdButton1_Click في Immediate واضغط Enter. يشغل VBA إجراء الحدث ويعرض الرسالة الخاصة به.

استدعاء إجراء من إجراء أخر

تشغل إجراء فرعياً من إجراء أخر باستخدام بناء الجملة نفسه الذي استخدمته بتشغيل وظيفة "عندما يتم تجاهل قيمة الإرجاع" يعتمد بناء الجملة الذي تستعمله على ما إذا كنت تستخدم كلمة أساسية Call. إذا استخدمت الكلمة الأساسية Call لاستدعاء إجراء فرعي يتطلب وسائط يجب أن تضمن قائمة الوسيطة في أقواس وإذا لم يتطلب الإجراء الفرعي وسائط، يمكن أن تضمن أو تحذف الأقواس:

```
Call subroutinename (argumentlist)
subroutinename argumentlist
```

استدعاء إجراء فرعي من تطبيق أخر

يمكنك استخدام طريقة Run لتشغيل إجراء فرعي من تطبيق أخر خلال procedure. استخدم بناء الجملة.

```
Application.Run subroutinename[, argument1, argument2, ...,
argumentN]
```

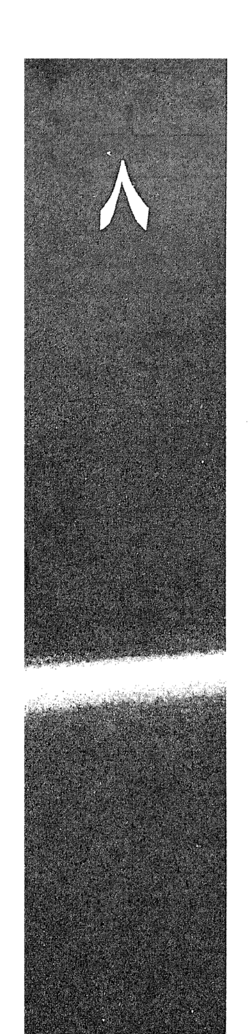
خلاصة

عرفك هذا الفصل ببعض أساسيات كتابة شفرة Access VBA. والنقاط الهامة هي:

- Access VBA له مجموعته الخاصة من أنواع البيانات بينما من المستحيل تجنب تعيين أنواع بيانات بنفسك بجعل VBA يعين نوع Variant لكل المتغيرات، يشتغل البرنامج الخاص بك بسرعة أكبر عندما تحدد أنواع بيانات معينة.
- Access VBA له نوع بيانات Object يمكنك استخدامه لمتغيرات كائن تشير لكائنات بالإضافة إلى ذلك، هناك أنواع فرعية لكائنات تشغيل البيانات ولأكسس.
- هناك نوعان من الوحدات. تخزن الوحدات القياسية إجراءات غير مرتبطة بكائن محدد. أما وحدات الفئة Class فهي تخزن تعريفات لكائنات جديدة هناك نوعان من وحدات الفئة وهما وحدات الفئة المستقلة وهي غير مرتبطة بنموذج أو تقرير وتخزن ككائنات منفصلة في قاعدة البيانات أما وحدات النموذج أو التقرير مخزن كجزء من النموذج أو التقرير.

- هناك نوعان من الإجراءات يمكن لإجراءات الوظيفة أن ترجع قيمة. تستخدم إجراءات الوظيفة في التعبيرات يشغل VBA الإجراء عندما يتم تقييم الإجراء. عندما تشغل إجراء وظيفة باستخدام تقنية استدعاء أخرى، يتم تجاهل قيمة إرجاع الوظيفة. لا ترجع الإجراءات الفرعية قيمة ما تخزن الإجراء الفرعي في وحدة النموذج أو التقرير وتشغل الإجراء عندما يتعرف كائن في النموذج أو التقرير على حدث.
- تتضمن بيئة برمجة Access VBA إطار Module الذي تدخل فيه عبارات شفرة كمجموعة من أشرطة الأمر المضمنة بأوامر للإنشاء والتحرير واكتشاف أخطاء الشفرة وتصحيحها.
- يمكنك حفظ قاعدة البيانات كملف mde. الذي تزال فيه مصدر شفرة Visual Basic ويبقى الإصدار المجمع فقط. لا يمكن تغيير النماذج والتقارير والوحدات في ملف mde.
- يمكنك تشغيل إجراء الوظيفة باستدعائها في التعبيرات واستدعائها من إجراءات أخرى أو استدعائها من ماكرو أو تطبيق أخر باستخدام Automation، مطلقاً لهم بالأحداث واستدعاؤهم في إطار Immediate.
- يمكنك تشغيل الإجراءات الفرعية باستدعائها من إجراءات أو تطبيق أخر باستخدام "Automation" مطلقاً لهم بالأحداث واستدعائهم في إطار Immediate.

يستكمل الفصل التالي أساسيات Access Visual Basic بتعريفك باستخدام بعض المتغيرات والثوابت في الإجراءات.



استخدام التغييرات

في الفصل السابق تعلمت أساسيات إنشاء إجراءات وتخزينهم في وحدات نمطية وفي هذا الفصل تكمل وضع أساسيات إنشاء إجراءات VBA عن طريق استخدام المتغيرات في الإجراءات ويعتبر استخدام المتغيرات هو أساس إنشاء تعليمات برمجية يعاد استخدامها.

مفهوم المتغير هو: المتغير هو موقع يسمى في الذاكرة وهو إما يحتوي على قيمة أو يشير إلى كائن ومع ذلك فإن تنفيذ هذا المفهوم في أكسس VBA معقد بعض الشيء بسبب المرونة الهائلة التي يوفرها استخدام المتغيرات. نحاول في هذا الفصل اختبار كيفية إنشاء متغير ومشاركته مع إجراءات أخرى وكيفية تمرير المتغيرات عند استدعاء إجراء وكيفية التخلص من هذا المتغير عند الانتهاء من استخدامه كذلك لأن المتغيرات يتم الاحتفاظ بها في الذاكرة فستحاول توضيح تأثير المتغيرات على أداء واستخدام الذاكرة. بعد الانتهاء من مناقشة المتغيرات يغطي الفصل إنشاء ثوابت مخصصة واستخدام الصفوف للعمل مع عدة متغيرات في نفس الوقت وفي الجزء الأخير من الفصل تتعلم كيفية إنشاء أنواع البيانات الخاصة بالمستخدم.

استخدام المتغيرات في الإجراءات

تعتبر كل الإجراءات التي أنشئت في الفصل السابق والموجودة على قاعدة بيانات Ch7_Examples بسيطة النوعية.

- تعتبر عبارات التعليمات البرمجية التي كتبت بسيطة لأن هدفها هو تعليم المستخدم كيفية إنشاء وتخزين وتشغيل إجراءات "سنرجى" التعامل مع العبارات الأكثر عملية إلى فصول أخرى.
- كان استخدام المتغيرات محدود للغاية لأن الإجراءات استخدمت المتغيرات كوسائط فقط. عند تشغيل الإجراءات مع الوسائط تحدد قيم حرفية للوسائط مثل Concatenate ("Johnson", "Diane") و Function1(3).
- بسبب عدم الإعلان صراحة عن أنواع البيانات تكون قيم إرجاع الوسائط والقيم التي حددناها والدالة أشكال مختلفة كما شرحنا في الفصل السابق يستخدم نوع بيانات الأشكال المختلفة Variant أكثر من ضعف الذاكرة التي يستخدمها أي نوع بيانات أساسي أخر ويتسبب ذلك في بطئ الأداء من لو قمنا بتعريف نوع بيانات محدد.

الهدف من هذا الجزء هو تعلم كيفية استخدام المتغيرات في الإجراءات والسببين الرئيسين لاستخدام المتغيرات هما.

- إنشاء تعليمات برمجية يُعاد استخدامها إذا أمكن إعادة استخدام الإجراءات بذلك تقل كمية التعليمات البرمجية التي تكتب.
- إنشاء تعليمات برمجية سريعة.

يزيل مثالين بسيطين أي شك في كيفية استخدام المتغيرات في الإجراءات.

ملاحظة لإتقان المفاهيم التي نستكشفها في هذا الفصل قم بإنشاء نسخة من قاعدة البيانات Northwind المسماة Northwind_Ch8 تأكد من اختيار خيار Always Use Event Procedures في علامة تبويب Forms/Reports Tab من مربع Options، يظهر مربع Options بعد اختيار الأمر Tools ⇦ Options.

استخدام المتغيرات لإنشاء تعليمات برمجية يعاد استخدامها

لنفترض أن المستخدم يرغب في إنشاء نموذج لوحة تبديل Switchboard لها أزرار أوامر لفتح نماذج أخرى وإخفاء لوحة التبديل الاتجاه الأول لتنفيذ ذلك هو وضع أزرار الأوامر على لوحة التبديل الجديدة وإنشاء إجراءات أحداث لتنفيذ الإجراءات كمثال لذلك، تقوم بتنفيذ الخطوات اللازمة لفتح نموذج Customers في تطبيق Northwind.

١- أنشئ نموذج جديد في Northwind_Ch8 يسمى Frmswitchboard.

٢- ضع زر أمر على النموذج وقم بإعداد خاصية Name إلى Cmdcustomers وخاصية Caption إلى Customers.

٣- انقر داخل خاصية OnClick وانقر زر Build إلى يمين مربع الخاصية يقوم اكسس بإنشاء وفتح الوحدة النمطية للنموذج ويعرض قالب التعليمات البرمجية لإجراء الحدث.

٤- ادخل إجراء الحدث الموضح فيما يلي:

```
Private Sub Cmdcustomers_Click()
    Docmd.Openform "Customers"
    Forms!Frmswitchboard.Visible = False
End Sub
```

يقوم إجراء Cmdcustomers_Click بتشغيل طريقة Openform لكائن DOCMD لفتح النموذج. يحدد الإجراء اسم نموذج وسيطة طريقة OPENFORM كقيمة حرفية هي "Customers" لإخفاء لوحة التشغيل يقوم الإجراء بإعداد خاصية النموذج Visible إلى False ودون وجود مهام أخرى يقوم بها الإجراء فإنه ينتهي.

٥- اضغط النموذج، قم بالتبديل إلى طريقة عرض View وانقر الزر، يتيح نموذج Customers وتختفي لوحة التبديل.

الآن تقوم بتحسين الإجراء.

استخدام خاصية ME لتحسين الأداء

في إجراء Cmdcustomers_Click من أجل تشغيل العبارة التي تخفي لوحة التبديل يجب أن يتعامل اكسس مع مرجع هيكلي لخاصية Visible.

Forms!Frmswitchboard.Visible

يحتاج كل مستوى من المرجع وقت ليتم معالجته وهذا يعني أن كل علامة تعجب وكل نقطة تمثل وقت تنفيذ مطلوب لمعالجة المرجع ويمكن تجنب الهيكلية في هذه العبارة باستخدام خاصية ME تعتبر تلك الخاصية إحدى الأدوات المثلى الخاصة التي يوفرها برنامج اكسس عند إنشاء إجراءات في وحدة نمطية للنموذج أو التقرير استخدم خاصية ME في إجراء في وحدة نمطية لنموذج أو تقرير هي أسرع وسائل الإشارة إلى النموذج أو التقرير لأن VBA لا يحتاج إلى استهلاك وقت لمعالجة مرجع مؤهل كاملاً أول خطوة في هذا الإجراء لتحسين الأداء هي استبدال Forms!Frmswitchboard بخاصية ME.

استخدم (الوسيط) Argument بدلاً من الحرفي Literal

يوجد في نموذج Switchboard عدة أزرار تفتح نماذج مختلفة بدلاً من إنشاء إجراء فردي يمكن أن يعاد استخدامه أول خطوة تنفذ لجعل إجراء موجود بالفعل يعاد استخدامه هي تحديد ما الذي يمنع إمكانية إعادة استخدامه في إجراء Cmdcustomers_Click تكمن المشكلة في ظهور "Customers" حرفياً في الوسيطة الخاصة بأسلوب OPENFORM تقوم بإخراج الحروف من الإجراء باستبدال الحرفي في وسيطة Openform سلسلة متغيرات والتي ستطلق عليها اسم Strformname استخدم علامة اسم STR لأننا نهتم بأنواع البيانات ثم بعد ذلك نرتب لتمرير الحرفي مرة أخرى إلى الإجراء كوسيطة للإجراء استخدم بناء الجملة التالي لتعريف نوع البيانات لوسيطة.

```
Procedurename (Argumentname [As Datatype])
Using A "Typed" Argument, The Procedure Looks Like This:
Private Sub Cmdcustomers_Click(Strformname As String)
    Docmd.Openform Strformname
    Me.Visible = False
End Sub
```

التغيير إلى إجراء دالة

الإصدار المعدل للإجراء الذي تقوم بتنفيذه لم يعد إجراء حدث لأنه وفقاً للتفريق فإن إجراء الحدث Click ليس له أي وسائط إذا حاولت تجميع الإجراء الفرعي حتى لا يستخدم بناء الجملة التي يحتفظ به لإجراء حدث ويتم استدعاء الإجراء الفرعي من إجراء حدث جديد ومع ذلك لا يعتبر ذلك حل جيد لأننا سنقوم فقط بتمرير المشكلة الحرفية إلى إجراء الحدث الجديد. إذا تم إعادة تسمية الإجراء الفرعي إلى Openaform وتم استدعائه من إجراء حدث، يظل الحرفي جزء من تعليمات البرمجية في VBA.

```
Private Sub Cmdcustomers_Click()
   Call Openaform ("Customers")
End Sub
```

القيمة الحرفية التي تم إدخالها في إطار Module تسمى قيمة تعليمات برمجية مدخلة يدوياً Hard-Coded Value وهي غالباً ما تمنع التعليمات البرمجية من إعادة استخدامها.

الحل الأفضل هو تغيير الإجراء إلى إجراء دالة يسمى Openaform وذلك ليمكن إدخال الحرفي في ورقة الخاصية بدلاً من كونه قيمة مدخلة يدوياً في الوحدة النمطية. عند إنشاء إجراء دالة، يقوم برنامج VBA بصورة آلية بوضع موقع الذاكرة جانباً لقيمة الإرجاع سواء كانت الدالة ترجع أم لا وفي حالة عدم تحديد نوع بيانات لهذا الموقع فإن برنامج سيفرض استخدام نوع بيانات Variant ولتجنب استخدام نوع بيانات Variant الذي يقلل من مستوى الأداء يرجع قيمة أو الدالة كأنها عدد صحيح Integer. استخدم بناء الجملة التالي لتحديد نوع البيانات لقيمة إرجاع إجراء الدالة:

```
[Public|Private] Function Functionname [(Argumentlist)] [As Type]
```

يمكن الآن إعادة استخدام إجراء الدالة الموضح فيما يلي بواسطة أزرار أوامر أخرى على نفس النموذج.

```
Public Function Openaform (Strformname As String) As Integer
   Docmd.Openform Strformname
   Me.Visible = False
End Sub
```

استخدم بناء الجملة التالي لتعيين إجراء الدالة Click لأمر على النموذج.

```
=Openaform("Formname")
```

الخطوة التالية هي جعل إجراء الدالة يعاد استخدامه على أي نموذج وكذلك تقوم بتخزين إجراء الدالة يعاد استخدامه على أية نموذج وكذلك تقوم بتخزين إجراء الدالة في وحدة نمطية

قياسية لأن إجراء دالة يعاد استخدامه لا يكون مرتبط بنموذج محدد ومن نتائج نقل الإجراء إلى وحدة نمطية قياسية هو عدم قدرة المستخدم أن يستخدم مرجع ME حيث يمكن استخدام مرجع ME الإشارة إلى نموذج أو تقرير فقط في حالة تخزين الإجراء في الوحدة النمطية للنموذج أو للتقرير للاحتفاظ بالإجراء ليعاد استخدامه: يمكن استخدام كائن Screen للإشارة إلى النموذج الذي يوجد به زر المر ولإخفاء النموذج قبل فتح نموذج Customers في حالة عدم تغيير ترتيب العبارات، يفتح الإجراء ثم يغلق النموذج المحدد لأن النموذج المفتوح حديثاً يصبح النموذج النشط.

١- قم بإنشاء وحدة نمطية قياسية جديدة تسمى Basnavigation وإدراج دالة عامة جديدة تسمى Openaform ثم أدخل الإجراء الموضح فيما يلي:

```
Public Function Openaform (Strformname As String) As Integer
    Screen.Activeform.Visible = False
    Docmd.Openform Strformname
End Sub
```

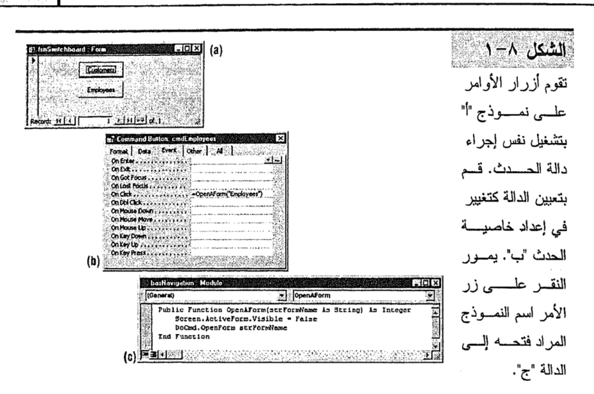
٢- افتح نموذج Frmswitchboard في طريقة عرض التصميم انقر في زر Customers لخاصية Onclick واستبدل [Event Procedure] بإجراء دالة حدث بوسيطة حرفية عن طريق كتابة =Openaform("Customers").

٣- قم بإعداد خاصية النموذج Hasmodule إلى No وانقر Yes في مربع التحقق من الصحة ثم احفظ النموذج وقم بالتبديل إلى طريقة عرض النموذج.

٤- انقر الزر. يقوم برنامج VBA بتشغيل إجراء دالة الحدث لإخفاء نموذج Frmswitchboard وافتح نموذج Customers وانسخ والصق الزر لفتح نموذج آخر راجع الشكل "٨-١ أ".

٥- افتح نموذج Frmswitchboard في طريقة عرض التصميم، انسخ والصق الزر ثم حدد الزر الملصوق وقم بتغيير خاصية Name إلى Cmdemployees وكذلك قم بتغيير خاصية Caption إلى Employees. انقر في خاصية Onclick وقم بتغيير الوسيطة لدالة الحدث إلى "Employees" راجع الشكل "٨-١ ب".

٦- احفظ النموذج، قم بالتبديل إلى طريقة عرض النموذج وانقر زر Employees يقوم برنامج أكسس بتشغيل إجراء دالة الحدث ليوضح الشكل "٨-١ ج" الإجراء في إطار Module.



الشكل ٨-١

تقوم أزرار الأوامر على نموذج "أ" بتشغيل نفس إجراء دالة الحدث. قم بتعيين الدالة كتغيير في إعداد خاصية الحدث "ب". يمرر النقر على زر الأمر اسم النموذج المراد فتحه إلى الدالة "ج".

كخطوة أخيرة لجعل إجراء دالة Openaform يعاد استخدامه، يجب استخدام اسم النموذج كعنوان تسمية لزر الأمر في هذه الحالة، يحدد الإجراء خاصية Caption لزر الأمر ولا تحتاج لتمرير اسم النموذج كوسيطة وتكون الدالة المعدلة هي:

```
Public Function Openaform () As Integer
   Dim Strformname As String
   Strformname = Screen.Activecontrol.Caption
   Screen.Activeform.Visible = False
   Docmd.Openform Strformname
End Sub
```

تكون خاصية Onclick هي =Openaform().

تلميح من فوائد استخدام إجراء دالة حدث عندما يكون الكائن قد تم نسخه ولصق التعيينات إلى إجراءات دالة الحدث ويتم تجاهل تعيينات الإجراءات الفرعية للحدث.

استخدام المتغيرات لتعليمات برمجية سريعة

لنفترض وجود نموذج يراد استخدامه لكل من المراجعة ولإدخال البيانات. في وضع إدخال البيانات العادي تتم إتاحة عناصر تحكم البيانات وتكون غير مؤمنة ولها الخلفية البيضاء القياسية

بالنسبة لوضع المراجعة يجب إلغاء إتاحة وتأمين كل عناصر تحكم البيانات لمنع تغييرات البيانات غير المقصودة ويتم تغيير لون الخلفية ليطابق لون خلفية النموذج كدليل مرئي على أن البيانات لا يمكن تغييرها. ما يلي هي الخطوات الواجب اتباعها لتغيير خصائص عنصر تحكم:

١- قم بإنشاء نموذج جديد يسمى Frmdots ثم ضع زر أمر وتكون خاصية Name هي Cmdchange وخاصية Caption هي Change مع وجود عنصر تحكم مربع نص يسمى Txtchange على النموذج وتأكد من وجود زر الأمر في أول التبويب ليكون عليه التركيز عند فتح النموذج. يجب أن يكون لزر الأمر خاصية Tab Index قيمتها 0 ويجب كذلك أن يكون لمربع النص Tab Index قيمته 1. في الخطوة التالية، نقوم بإنشاء إجراء ليمنع إتاحة مربع نص Txtchange ولأنه لا يمكن عدم إتاحة عنصر تحكم يكون التركيز عليه لذلك لا يمكن لعنصر تحكم Txtchange أن يحصل على التركيز عند تشغيل الإجراء.

٢- في الوحدة النمطية القياسية Basnavigation، ادرج إجراء Change الموضح فيما يلي، يقوم إجراء CHANGE بتغيير خصائص مربع نص TXTCHANGE:

```
Public Function Change ()
    Forms!Frmdots!Txtchange.Enabled = False
    Forms!Frmdots!Txtchange.Locked = True
    Forms!Frmdots!Txtchange.Backcolor = 12632256
End Sub
```

٣- انقر في خاصية Onclick لزر الأمر واكتب ()Change = ثم احفظ النموذج وقم بالتبديل إلى طريقة عرض النموذج راجع الشكل "٨-٢ أ".

٤- انقر زر Change وبذلك تتغير خصائص عنصر تحكم مربع النص.

لنقوم بتحسين الإجراء

الحد من نقاط علامة التعجب والنقاط

كما وضحنا فيما سبق فإن كل نقطة علامة تعجب وكل نقطة تمثل وقت تنفيذ يستغرق معالجة المرجع. يوجد في إجراء Change تسع نقاط علامة تعجب ونقاط وينتج عن كل نقطة علامة تعجب ونقطة يتم الحد منها الإسراع في التعليمات البرمجية. بالنسبة لإجراء واحد فإن عدد نقاط علامة التعجب والنقاط لا يهم كثيراً ولكن لنفكر قليلاً في عدد النقاط ونقاط علامة التعجب المحتملة في تطبيق آلي. ولهذا يجب أن يتم حساب عدد النقاط وتصغير حجمها ولكن كيف يتم ذلك مع المتغيرات.

في إجراء Change فإن الكائن المشار إليه هو عنصر تحكم مربع نص ولذلك يتم إنشاء متغير كائن جديد لتمثيل مربع النص. لإنشاء متغير في إجراء يمكن استخدام عبارة مع بناء الجملة التالي:

```
Dim Variablename [As Type]
```

تسمى هذه العبارة "عبارة تعريف متغير" وتجعل أكسس يقوم بوضع موقع الذاكرة جانباً وتعيين اسم له هو Variablename. تحدد كمية الذاكرة التي تم وضعها جانباً بواسطة جزء AS Type الموجود في العبارة وهذا الجزء اختياري ويقوم أكسس بتعيين نوع بيانات Memory-Hungry Variant إلا إذا تم تحديد نوع بيانات آخر. وفي تلك الحالة، الكائن هو عنصر تحكم مربع نص ولذلك نستخدم نوع بيانات كائن Textbox:

```
Dim Ctl As Textbox
```

تنشئ تلك العبارة Ctl كمتغير كائن من نوع Textbox ويوضع موقع الذاكرة جانباً ولكنه لا يحتوي على شيء وهي قيمة Nothing إلى أن يتم تعيين كائن بعبارة مهمة:

```
Set Ctl = Screen.Activeform.Txtchange
```

نستخدم هنا كائن Screen للإشارة إلى النموذج النشط وننهي الرجوع إلى نموذج محدد. نستطيع الآن استبدال المرجع الهيكلي لعنصر التحكم مع المتغير. ويصبح إجراء Change كما يلي:

```
Public Function Change ()
    Dim Ctl As Textbox
    Set Ctl = Screen.Activeform.Txtchange
    Ctl.Enabled = False
    Ctl.Locked = True
    Ctl.Backcolor = 12632256
End Sub
```

قم بحساب عدد نقاط علامات التعجب النقاط يوجد الآن فقط خمسة نقاط وبذلك يعمل الإجراء أسرع مما لو كان هناك تسع نقاط.

إنشاء ثابت مخصص

أخر تعديل يتم في الإجراء هو التعامل مع الرقم الذي يستخدمه أكسس اللون الرمادي عن طريق إنشاء ثابت مخصص يمكن عن طريق تعريف ثابت مخصص تجنب الحاجة إلى كتابة الرقم مجدداً ويمكن إنشاء ثابت مخصص في إجراء باستخدام عبارة التعريف الثابتة التالية:

```
Const Constantname [As Type] = Value
```

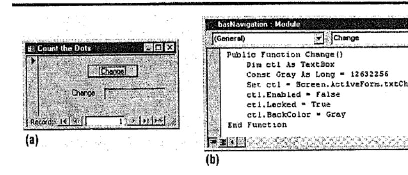
مثل عبارة تعريف متغير، تضع عبارة تعريف ثابت موقع الذاكرة جانباً وتطلق عليـــه اسـم محدد ويستخدم جزء As Type الموجود في العبارة لتعريف نوع بيانات الثابت حتـــى يستطيــع أكسس تحديد كمية المساحة المطلوب تحديدها لهذا الثابت. الفرق بين عبـــارة تعريـــف متغيـر وعبارة تعريف ثابت هي أنه بالفعل يتم استخدام عبارة تعريف الثابت لتعيين قيمة غير متغيـــرة. في هذا المثال، يكون الرقم عدد صحيح ولكنه يتعدى حدود نوع بيانات العدد الصحيــح Integer ولذا يتم تعريفه على أنه As Long كما يلي:

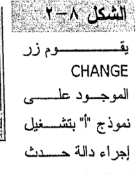
```
Const GRAY AS LONG = 12632256
```

الإجراء المحسن موضح فيما يلي وفي الشكل "٨-٢ ب" تتعلم في الفصل ٩ بعض التقنيـــات الإضافية (With...End With) لتحسين أداء هذا الإجراء.

```
Public Function Change ()
   Dim Ctl As Textbox
   Const Gray As Long = 12632256

   Set Ctl = Screen.Active.Form.Txtchange
   Ctl.Enabled = False
   Ctl.Locked = True
   Ctl.Backcolor = Gray
End Sub
```





الشكل ٨-٢ يقـــوم زر CHANGE الموجـــود علـــى نموذج "أ" بتشغيـــل إجراء دالة حـــدث في الشكل "ب".

كيف تستخدم متغيرات الإجراءات

يوضح المثالين السابقين في الجزء الأخير بعض أساليب استخدام الإجراءات للمتغيرات والثوابت. يمكن للإجراء استخدام المتغيرات والثوابت بهذه الطرق:

- يمكن إنشاء متغيرات وثوابت لاستخدامها وتكون الثوابت والمتغيرات التي يتم إنشائها داخل إجراء متاحة فقط للإجراء الذي تم إنشائهم بداخله ولا يكونوا متاحين لأي إجراء آخر حتى ولو كانت تلك الإجراءات في نفس الوحدة النمطية. تسمى الثوابت والمتغيرات التي يتم إنشائها داخل إجراء محلي Local أو مستوى إجراء Procedure-Level.
- يمكن استخدام المتغيرات كوسائط. قد يطلب الإجراء معلومات إضافية كوسائط من أجل تحديد كيف يعمل الإجراء. يمكن للمعلومات الإضافية في نموذج وسائط المتغير أن تعطي أو يتم تمريرها إلى الإجراء عند استدعائه.
- يمكن أن تستخدم متغيرات وثوابت تم إنشائها في مكان آخر. يمكن إنشاء متغيرات وثوابت عامة في جزء Declarations في الوحدة النمطية وتكون تلك الثوابت والمتغيرات العامة متاحة لبعض أو لكل الإجراءات في قاعدة البيانات. تسمى المتغيرات والثوابت التي يتم إنشائها في جزء Declarations في الوحدة النمطية مستوى الوحدة النمطية Module-Level.
- يمكن أن تنفذ عبارات تقوم بتعيين قيمة المتغيرات التي يستطيع الإجراء الوصول إليها.
- يمكن أن ترجع متغير. يمكن لإجراء دالة أن يرجع متغير.

يصف الجزء التالي الطرق المختلفة لإنشاء والتخلص من متغيرات وكيفية تمرير متغيرات من إجراء إلى آخر.

تعريف المتغيرات

يمكن إنشاء كلا من الثوابت والمتغيرات باستخدامهم في العبارات. أول مرة يتم استخدام اسم متغير أو ثابت يقوم VBA بصورة آلية بإنشاء موقع تخزين مؤقت في الذاكرة بالاسم الذي تم تحديده وهذا يسمى تعريف ضمني Implicit Declaration. هناك مشكلة واحدة مع التعريفات الضمنية وهي أن VBA لا يتعرف على أخطاء الطباعة ويقوم ببساطة بإنشاء متغير جديد في حالة الخطأ في كتابة اسم بدون الربط بين الاسمين. يمكن تجنب مصدر هذا الخطأ غير الضروري بإضافة عبارة Option Explicit في جزء Declaration لكل وحدة نمطية وذلك لطلب أن يتم إنشاء كل المتغيرات والثوابت بصورة واضحة في عبارات التعريف. عبارات التعريف العادية هم العبارتين اللتين يتم وضعهم في بداية إجراء Change الموضح فيما سبق:

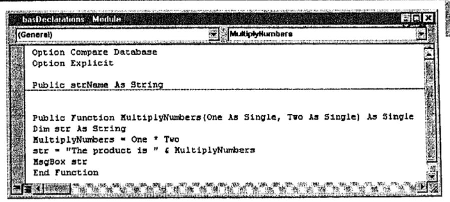
```
Dim Ctl As Textbox
Const Gray As Long = 12632256
```

ملاحظة يفترض هذا الفصل أنه قد تم تعريف كل الثوابت والمتغيرات بصورة واضحة وأن عبارة OPTION EXPLICIT تستخدم في جزء DECLARATIONS لكل وحدة نمطية. يمكن أن تجعل VBA يقوم بصورة آلية بإدخال عبارة OPTION EXPLICIT عن طريق اختيار خانة REQUIRE VARIABLE DECLARATION في تبويب EDITOR الموجود على مربع OPTIONS الخاص بمحرر فيجوال بيسك والذي يتاح عن طريق اختيار TOOLS ⇦ OPTIONS في محرر فيجوال بيسك. هذا الخيار لا يتم تحديده بصورة افتراضية.

يوجد مكانين فقط يمكن فيهما إنشاء متغير:

- وضع عبارة تعريف داخل إجراء أو في قائمة وسائط الإجراء لإنشاء متغير محلي أو متغير مستوى الإجراء.
- وضع عبارة تعريف في جزء DECLARATION لوحدة نمطية لإنشاء متغير مستوى الوحدة النمطية.

يوضح الشكل "٨-٣" متغير مستوى الإجراء ومتغير مستوى الوحدة النمطية.



الشكل ٨-٣ متغير مستوى الإجراء STR ومتغير مستوى الوحدة النمطية STRNAME.

عند إنشاء متغير أو ثابت يجب تحديد السمات التالية:

- الاسم.
- نوع بيانات الثابت أو المتغير العادي أو نوع بيانات الكائن لمتغير كائن.
- مدة حياة المتغير وهي المدة التي يستغرقها التنفيذ بين إنشاء ثابت أو متغير وعندما لا يوجد.

- مجال الثابت أو المتغير "ويحدد المجال أي الإجراءات التي يمكنها رؤية واستخدام الثابت أو المتغير الخاص أو العام".

تسمية المتغيرات والثوابت

يتم استخدام عبارات التعريف لتحديد اسم الثابت أو المتغير. ولا تهتم أسماء VBA بحالة الأحرف سواء كانت أحرف كبيرة أو صغيرة وهذا يعني أن VBA لا يستطيع التمييز بين Frmcustomers وFrmcustomers. ومع ذلك، فبعد إنشاء متغير بصورة واضحة يغير VBA بصورة آلية التواجد التالي للاسم ليطابق حالات الأحرف الصغيرة والكبيرة التي يتم تحديدها في عبارة التعريف.

يجب أن تبدأ أسماء الثوابت والمتغيرات بحرف، تحتوي فقط على حروف أرقام وعلامة التسطير أسفل السطر (_) ولا تحتوي على كلمات أساسية ولا تتعدى الأحرف ٢٥٥ حرف. يناقش الفصل ٢ ابتهاج أسلوب التسمية للمساعدة في جعل تعليمات برمجة VBA أسهل في الفهم والقراءة.

تحديد نوع بيانات

يمكن استخدام عبارة As Type في عبارة التعريف لتعريف نوع بيانات المتغيرات. بينما يعتبر تعريف نوع البيانات اختياري فإن عدم تحديد نوع بيانات يجعل VBA يعين نوع بيانات Variant. كما يوضح الجدول ١-٧ في الفصل ٧ يستخدم نوع بيانات Variant أكثر من ضعف الذاكرة لأي نوع بيانات أساسي آخر. لأن الذاكرة دائماً محدودة فإنه كلما كثرت مساحة الذاكرة المستخدمة في تخزين المتغير كلما انخفضت المساحة المتاحة لأكسس والتطبيقات الأخرى التي يتم تشغيلها. مع وجود مساحة قليلة في الذاكرة يعمل أكسس بصورة أكثر بطأ ولذا يجب تجنب استخدام نوع بيانات Variant على أساس أن عدم استخدامه هو الأنسب لإدارة الذاكرة بصورة جيدة. مع ذلك هناك أسباب أداء إضافية لتجنب استخدام نوع بيانات Variant:

- في كل مرة يقوم إجراء بتعيين قيمة لمتغير مع نوع بيانات Variant يجب أن يأخذ VBA وقت تنفيذ لتحديد نوع البيانات التي تتصل بالقيمة ثم يغير نوع البيانات للمتغير ليطابق البيانات وتسمى تلك العملية Coercing متغير التباين.
- عند رؤية متغير تباين Coerced في عملية حسابية يحتاج VBA إلى أخذ وقت تنفيذ إضافي لتحويل نوع البيانات من أجل تنفيذ العملية الحسابية إذا كان VBA لا يستطيع تحويل نوع البيانات، يحدث خطأ وقت التشغيل.

- ♦ لتجنب أخطاء وقت التشغيل تحتاج إلى تضمين عبارات للتحقق من نوع بيانات المتغير. على سبيل المثال، في حالة القيام بعملية حسابية عددية يمكن تجنب الأخطاء باستخدام دالة Isnumeric لتحديد إذا كان للمتغير نوع بيانات عددي قبل إجراء العملية الحسابية. يأخذ تنفيذ التعليمات البرمجية للتحقق من نوع البيانات وقت تنفيذ إضافي وفي حالة استخدام نوع بيانات محدد بدلاً من ذلك يمكن أن يتحقق VBA من توافق نوع البيانات بعد تجميع التعليمات البرمجية.

إلا إذا كان هناك سبب محدد لاستخدام نوع بيانات Variant يجب كتابة المتغيرات بصورة واضحة. بالنسبة للمتغيرات العادية فهذا يعني اختيار نوع البيانات الذي يتطلب أصغر حجم ذاكرة من قائمة أنواع البيانات الأساسية المدرجة في جدول ١٧.

بالمثل بالنسبة لمتغيرات الكائن يجب تجنب استخدام نوع بيانات Object العام. يعتبر استخدام نوع بيانات Object لمتغير كائن مثل استخدام نوع بيانات Variant لمتغير عادي في أنه في كل مرة يقوم الإجراء بتعيين كائن يجب أن يأخذ VBA وقت تنفيذ لتحديد نوع الكائن الذي تم تعيينه وفي كل مرة يشير فها الإجراء إلى خاصية أو إلى طريقة يأخذ VBA وقت تنفيذ لتحديد إذا كانت الخاصية أو الطريقة صحيحة بالنسبة للكائن. في حالة استخدام نوع بيانات كائن محدد بدلاً من ذلك يتحقق VBA من صحة الخاصية والطريقة عند تجميع البيانات. بالنسبة لمتغير كائن فإن استخدام نوع بيانات كائن محدد يسمى الكتابة اليدوية Hard Typing.

ملاحظة للحصول على أكفأ إدارة للذاكرة وأسرع تعليمات برمجية، استخدم أكثر أنواع البيانات تحديداً للمتغيرات العادية وأكثر أنواع الكائنات تحديداً لمتغيرات الكائن.

فهم دورة حياة متغير

عندما يقرأ VBA عبارة التعريف لمتغير مثل:

```
Dim Intx As Integer
Dim Txt As Textbox
```

فإنه ينشئ ويسمي موقع الذاكرة المؤقت ويحدد كمية التخزين المحددة بواسطة نوع البيانات الذي سبق تحديده ويقوم بإعداد المتغير إلى قيمة افتراضية بناء على نوع البيانات. يتم إعداد أي متغير له نوع بيانات رقمي إلى صفر، أما المتغير له نوع بيانات سلسلة يتم إعداده إلى سلسلة طولها صفر ("") ومتغير له نوع بيانات Variant يتم إعداده إلى قيمة Empty الخاصة وبالنسبة لمتغير كائن فيتم إعداده إلى قيمة Nothing الخاصة. تعيين عبارات التعريف الموضحة في بداية

هذا الجزء المتغير Intx إلى صفر ومتغير الكائن Txt إلى Nothing. أما ما يحدث بعد ذلك فيعتمد على ما إذا كان المتغير متغير عادي أو متغير كائن.

دورة حياة متغير عادي

بعد إنشاء متغير عادي يمكن التعامل مع هذا المتغير في العبارات. تبدأ باستخدام عبارة محددة لتعيين قيمة لاسم المتغير مثل Intx =، بعد تعيين المتغير يستخدم اسم المتغير في عبارات أخرى لتغيير القيمة مثل عبارة Intx = Intx + 1 مما يزيد قيمة Intx بواحدة وتعيين النتيجة إلى Intx.

يمكن كذلك استخدام عبارة تعيين لإعادة تجهيز متغير عن طريق إعداد المتغير إلى قيمته الافتراضية. بالنسبة للمتغيرات العادية مع نوع بيانات محدد، تجهيز المتغير يعني ببساطة إعداد قيمته إما إلى صفر أو إلى سلسلة طولها صفر. على سبيل المثال، العبارة Intx = 0 تعيد تجهيز متغير Intx.

في النهاية، يتم التخلص من المتغير نفسه. يعني التخلص من متغير التخلص من موقع الذاكرة المؤقت وترك الذاكرة ليعاد استخدامها أما كيفية التخلص من متغير فإن الإجابة ستأتي فيما بعد.

دورة حياة متغير كائن

بالنسبة لمتغير كائن، يصبح ذلك الإجراء معقد بعض الشيء لأن هناك عنصرين يجب متابعتهما وهما متغير الكائن والكائن نفسه. بعد تعريف متغير كائن استخدم عبارة تعيين ليشير متغير كائن إلى كائن مثل:

```
Set Txt = Forms!Frmdots!Txtchange
```

قد يوجد الكائن الذي يشير إليه المتغير في الذاكرة أو قد تنشئ عبارة التعيين الكائن في الذاكرة. على سبيل المثال عند فتح نموذج بصورة تفاعلية مثلما يحدث عند فتح نموذج Frmdots من إطار Database، يقوم أكسس بإنشاء كائن Form في الذاكرة. عند تعريف متغير كائن مثل Dim Frm As Form ويشير Frm إلى النموذج به عبارة التعيين Set Frm = Forms!Frmdots تجعل متغير الكائن يشير إلى كائن يوجد بالفعل في الذاكرة في المقابل عند تعريف متغير كائن Recordset مثل Dim Rst As Recordset فإنك تقوم بالفعل بإنشاء كائن Recordset جديد مع عبارة التعيين.

```
Set Rst = Currentdb.Openrecordset("Customers")
```

لأن هناك كلاً من متغير كائن وكائن، فهناك طلبين للذاكرة التي يستهلكها متغير الكائن والذاكرة التي يحتل مساحتها الكائن كما يشير جدول ١-٧ في الفصل ٧، يأخذ متغير الكائن مساحة في بايت مهما يكن ما يشير إليه قد يأخذ مئات أو آلاف من البايت في الذاكرة.

بعد تعيين متغير الكائن يستخدم متغير الكائن في عبارات أخرى لتغيير الكائن. عند تغيير متغير كائن في عبارة VBA فما يحدث فعلاً هو تغيير الكائن. على سبيل المثال، في إجراء Change فإن العبارات الثلاث التي تغير خصائص متغير كائن Txt تغير بالفعل خصائص مربع نص Txtchange الموجود في النموذج.

يمكن كذلك استخدام عبارة تعيين لإعادة تجهيز متغير كائن عن طريق إعداد متغير الكائن إلى قيمته الافتراضية. عند إعداد متغير كائن إلى Nothing فإنك تقوم بالتخلص من الرابطة بين متغير الكائن والكائن نفسه. بعد التخلص من الرابطة، فغن متغير الكائن مازال موجوداً وما زال يأخذ مساحة ٤ بايت من الذاكرة بينما قد يستمر أو لا يستمر الكائن في الوجود. على سبيل المثال، إذا كان FRM هو متغير الكائن الذي يشير إلى نموذج Frmdots وقمت بإعداد Frm = Nothing، يستمر كائن Form في الوجود في الذاكرة طالما ظل النموذج مفتوح. إعداد Frm إلى Nothing يؤدي دور الرابطة بين متغير الكائن والكائن. وفي المقابل إذا كانت RST هي متغير الكائن الوحيد الذي يشير إلى كائن Recordset الذي تم إنشائه مع عبارة التعيين:

```
Set Rst = Currentdb.Openrecordset("Customers")
```

يؤدي إعداد Rst إلى Nothing إلى الرابطة ويتخلص من الكائن. وفي هذه الحالة يتم التخلص من الكائن بسبب القاعدة التالية:

- ليظل الكائن متواجداً في الذاكرة يجب أن يتم تعيينه إلى متغير كائن أو يكون له مرجع ضمني إنشائه أكسس.

عند فتح النموذج، يقوم أكسس بصورة آلية بإنشاء مرجع ضمني لكائن Form الذي يستمر طالما النموذج مفتوح ولها تستطيع التعليمات البرمجية أن تؤثر في المرجع الضمني للنموذج طالما كان النموذج مفتوحاً. على عكس ذلك، عند فتح مجموعة سجلات فلا يوجد مرجع ضمني لكائن Recordset. طالما يوجد في التعليمات البرمجية متغير كائن واحد يشير إلى كائن Recordset، يظل كائن Recordset موجوداً في الذاكرة. وعند انفصال كل المراجع إلى كائن Recordset إما عن طريق إعداد متغيرات الكائن إلى Nothing أو عن طريق التخلص من متغيرات الكائن فإنه يتم التخلص من كائن Recordset نفسه وتترك مساحة ذاكرته تستخدم في أشياء أخرى.

يمكن التخلص من كائنات في الذاكرة عن طريق إغلاقهم. على سبيل المثال، يمكن إغلاق نموذج في إجراء باستخدام طريقة Close لكائن Docmd ويمكن إغلاق مجموعة سجلات باستخدام طريقة Close الخاصة بهم. إذا كانت التعليمات البرمجية تفتح قاعدة بيانات أخرى يمكن إغلاقها باستخدام طريقة Close الخاصة بها ولكن كاستثناء فإن استخدام طريقة Close على قاعدة البيانات الحالية وهي قاعدة البيانات المفتوحة في إطار Access لا يقوم بإغلاق تلك القاعدة. عند استخدام إحدى طرق Close على كائن يكون متغير الكائن الذي يشير إلى الكائن في

حالة إهمال: قد لا يوجد الكائن الذي كانت تشير إليه ولكن يظل المتغير نفسه موجوداً ويمكن تعيينه إلى كائن آخر.

التخلص من المتغيرات

بسبب مساحة الذاكرة التي تستهلكها المتغيرات لذلك يجب التخلص منهم بمجرد الانتهاء من استخدامهم. أما عن كيفية التخلص من متغير فإن أكسس لا يقدم عبارة لفعل ذلك ولذلك فلا يوجد طريق مباشر للتخلص من المتغير وبدلاً من ذلك فإن أكسس يقدم خيارين:

- يمكن تعريف متغير داخل إجراء كمتغير محلي أو متغير مستوى الإجراء. يقوم أكسس بالتخلص منه بصورة آلية بصورة افتراضية عند الانتهاء من الإجراء.
- يمكن تعريف متغير في جزء Declarations في الوحدة النمطية كمتغير مستوى الوحدة النمطية ويقوم أكسس بالتخلص منه عند إغلاق قاعدة البيانات وليس قبل ذلك.

نستعرض في الجزء التالي وجود أسباب صحيحة لاستخدام متغيرات مستوى الوحدة النمطية مع ذلك فإن القاعدة العامة واضحة بسبب مساحة الذاكرة التي تستهلكها المتغيرات، استخدم متغيرات مستوى الإجراء بدلاً من متغيرات مستوى الوحدة النمطية كلما كان ذلك ممكناً.

استخدام متغيرات مستوى الإجراء

هناك طريقتان لتعريف متغير في إجراء: في عبارة تعريف منفصلة داخل الإجراء أو في قائمة الوسيطة للإجراء الذي يتم استدعاء بواسطة إجراء آخر.

تعريف متغير داخل الإجراء

يمكن إنشاء متغير مستوى الإجراء عن طريق وضع عبارة تعريف داخل إجراء والعبارة الأساسية هي:

```
Dim Variablename [As Type]
```

ما يلي هو بعض الأمثلة:

```
Dim Intcounter As Integer
Dim Strlastname As String
Dim Frmcustomers As Form
```

يمكن تعريف متغيرات متعددة بعبارة تعريف واحدة ولكن يجب تضمين نوع بيانات كل متغير بصورة مستقلة: العبارة التالية تعرف Frm1 وfrm2 على أساس أنهم نوع كائن Form:

```
Dim Frm1 As Form, Frm2 As Form
```

لكن هذه العبارة تعرف Frm2 كنوع كائن Form و Frm1 كمتغير:

```
Dim Frm1, Frm2 As Form
```

تقوم عبارة التعريف بإنشاء وتسمية المتغير ويحدد موقع الذاكرة كما تم تحديده في عبارة As Type وتجهيز المتغير. لا تقوم عبارة التعريف بتعيين قيمة وتحتاج إلى عبارة تعيين منفصلة لتعيين قيمة. على سبيل المثال، راجع الإجراء التالي:

```
Public Sub Someprocedure()
   Dim Db As Database
   Set Db = Currentdb
   ...
End Sub
```

تنشئ عبارة التعريف التالية DB كمتغير كائن لنوع كائن قاعدة البيانات Database:

```
Dim Db As Database
```

توضح عبارة التعريف متغير الكائن لقاعدة البيانات الحالية:

```
Set Db = Currentdb
```

ملاحظة هناك إصدار آخر لعبارة التعريف الممكن استخدامها لمتغير كائن. يمكن استخدام الكلمة الأساسية New في تعريف متغير كائن باستخدام العبارة Dim Objvar As New Objecttype. عند تضمين الكلمة الأساسية New، يقوم VBA بإنشاء مثال جديد للكائن بصورة آلية ولا تحتاج استخدام عبارة Set لتعيين متغير كائن. راجع الفصل ١٤ لمزيد من المعلومات عن الكلمة الأساسية New.

تعريف متغير في قائمة الوسائط

عند وضع وسيطة في قائمة وسائط إجراء فإنك في نفس الوقت تنشئ متغير مستوى الإجراء. يمكن استخدام عبارة As Type لتحديد كل نوع متغير الوسيطة كما يلي:

```
Argumentname As Type
```

كمثال:

```
Public Function Concatenate (A As String, B As String)
```

عند تعريف متغير في قائمة وسائط، يحدد VBA بصورة آلية مساحة الذاكرة المعينة في عبارة نوع A ويجهز المتغير. مع ذلك، لا تحتاج إلى عبارة تعيين منفصلة لأن VBA يعين بصورة آلية قيمة للمتغير في قائمة الوسائط عند استدعاء الإجراء. على سبيل المثال، عند تنفيذ VBA للعبارة التالية:

```
Msgbox Concatenate ("Johnson","Diane")
```

يذهب Vba إلى دالة Concatenate ويعين بصورة آلية قيمة "Johnson" لمتغير A وقيمة "Diane" لمتغير B.

```
Public Sub Callingprocedure()
   Msgbox Concatenate ("Johnson","Diane")
End Sub
Public Function Concatenate(Lname As String, Fname As String)
   Concatenate = Lname & ", " & Fname
End Sub
```

فهم وضوح متغير مستوى الإجراء

يكون متغير مستوى الإجراء الذي تم إنشائه إما في عبارة تعريف أو في قائمة وسائط مرئياً فقط بالنسبة للإجراء الذي تم إنشائه فيه والإجراءات الأخرى لها تستطيع رؤية أو استخدام المتغير. الوسيلة الوحيدة التي يمكن لإجراء آخر الوصول بها إلى متغير مستوى الإجراء هي أن يقوم الإجراء الذي قام بإنشاء المتغير باستدعاء الإجراء الآخر ويمرر له المتغير كوسيطة. يمكن، في المقابل، لمتغير تم تمريره كوسيطة لإجراء تم استدعائه أن يتم تمريره إذا استدعى الإجراء الذي تم استدعائه أولاً إجراء آخر. يشار إلى سلسلة التتالي للإجراءات المستدعاه شجرة استدعاء الإجراء.

يوضح المثال التالي هذه الفكرة:

١– أنشئ وحدة نمطية جديدة تسمى Basvariables. ادرج إجراء فرعي عام جديد يسمى Localvariable كما يوضح ما يلي. يعرف إجراء Strlocal Localvariable كمتغير سلسلة مستوى الإجراء أو متغير محلي.

```
Public Sub Localvariable()
   Dim Strlocal As String
   Strlocal = "'Local Variable In Localvariable Procedure'"
   Msgbox Strlocal
End Sub
```

٢- قم بتشغيل الإجراء في إطار Immediate عن طريق طباعة Call Localvariable والضغط على Enter. الإجراء الوحيد الذي يمكن أن يستخدم متغير Strlocal هو إجراء Localvariable والإجراءات التي يقوم إجراء Localvariable بتمرير المتغير إليهم كوسائط.

٣- قم بإنشاء إجراء GETLOCAL الموضح فيما يلي إما في نفس الوحدة النمطية أو في وحدة نمطية أخرى وقم بتشغيله في إطار Immediate:

```
Public Sub Getlocal()
    Msgbox Strlocal
End Sub
```

يفشل الإجراء لأن إجراء GetLocal لا يرى متغير strLocal في إجراء LocalVariable. أما فيما يخص إجراء GetLocal فإن متغير strLocal لم يتم تعريفه.

٤- انقر زر Reset في شريط الأدوات. ادرج عبارة Call GetLocal بعد عبارة MsgBox في إجراء LocalVariable بعد ذلك قم بتشغيل إجراء LocalVariable في إطار Immadiate.

يعمل إجراء Localvariable ويطبع قيمة Strlocal ثم يفشل لوجود خطأ عدم تعريف متغير. يفشل الإجراء سواء تم استدعائه بصورة مستقلة أو من Localvariable لأن إجراء Getlocal لا يرى متغير Strlocal. لا يجعل مجرد استدعاء إجراء من إجراء له متغير محلي، المتغير المحلي متاح بالنسبة للإجراء الذي تم استدعائه. إذا أردت أن يستخدم الإجراء المستدعى المتغير يجب تمرير المتغير كوسيطة للإجراء المستدعى.

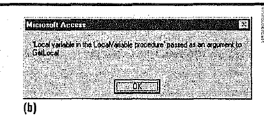
٥- قم بتعديل كلا الإجراءين كما موضح فيما يلي. مع هذه التغييرات يستدعي إجراء Localvariable. إجراء Getlocal ويمرر Strlocal كوسيطة ويستقبل إجراء Getlocal المتغير كوسيطة

```
Public Sub Localvariable()
    Dim Strlocal As String
    Strlocal = "'Local Variable In Localvariable Procedure'"
    Msgbox Strlocal
    Call Getlocal(Strlocal)
End Sub
Public Sub Getlocal(Stra As String)
    Msgbox Stra & " Passed As An Argument To Getlocal"
End Sub
```

٦- في إطار Immediate، اكتب Call LocalVariable واضغط Enter. يعرف ويعرض إجراء LocalVariable المتغير المحلي في مربع رسالة، راجع الشكل "٨-٤ أ"، ثم يستدعي إجراء GetLocal ويمرر متغير strLocal له كوسيطة. يستخدم إجراء GetLocal المتغير الذي تم تمريره إليه ويعرضه في مربع رسالة، راجع الشكل "٨-٤ ب".

ملاحظة يعتبر وضع عبارات تعريف لكل متغيرات مستوى الإجراء في بداية كل إجراء من إجراءات البرمجة الجيدة.

الشكل ٨-٤ يعرض الإجراء المتغير المحلي "أ" ثم يمرر المتغير المحلي لإجراء آخر الذي يقوم بدوره بعرض المتغير الذي تم تمريره إليه "ب".



تغيير مدة حياة متغيرات مستوى الإجراء

بعد تعريف متغير مستوى الإجراء وتعيين قيمة له في إجراء فإن عبارات أخرى قد تغير قيمته. ويتعامل المتغير مع القيمة الجديدة إلى أن تتغير مرة أخرى أو إلى أن ينتهي الإجراء.

افتراضياً، ينتهي وجود متغيرات مستوى الإجراء عند انتهاء استخدام الإجراء حيث يتخلص أكسس من قيمهم ويترك مساحات مواقع التخزين في الذاكرة ليعاد استخدامها. عموماً يمكن تطويل مدة حياة متغير مستوى الإجراء بعد فترة وجود الإجراء نفسه باستخدام الكلمة الأساسية Static بدلاً من الكلمة الأساسية DIM عند إنشاء المتغير. استخدم الكلمة الأساسية Static يطيل مدة حياة متغير إجراء لطوال مدة تشغيل تعليمات VBA البرمجية في أي وحدة نمطية. عند إنشاء متغير ثابت يحفظ VBA قيمته عند انتهاء الإجراء ويُعد المتغير إلى القيمة المحفوظة في المرة التالية التي يعمل فيها الإجراء.

كمثال، نقوم بإنشاء إجراء يستخدم متغير ثابت. ادرج إجراء Staticvariable الموضح فيما يلي في الوحدة النمطية Basvariables.

```
Public Sub Staticvariable()
    Dim Strnonstatic As String
    Static Sstrstatic As String
    Strnonstatic = Strnonstatic & " Nonstatic"
    Sstrstatic = Sstrstatic & " Static"
    Debug.Print Strnonstatic
    Debug.Print Sstrstatic
End
```

ينشئ إجراء Staticvariable متغير Strnonstatic كمتغير غير ثابت و sstrstatic كمتغير ثابت فهو يجهز كلاً من المتغيرات إلى سلسلة طولها صفر، يمكن استخدام حرف S بادئ للإشارة إلى متغير ثابت. تربط عبارة التعيين قيمة Strnonstatic مع كلمة غير ثابت Nonstatic وقيمة Sstrstatic مع كلمة ثابت Static ثم تطبع القيم المتسلسلة في إطار Immediate.

قم بتشغيل الإجراء في إطار Immediate عن طريق كتابة Callstaticvariable والضغط على Enter. أول مرة يتم تشغيل الإجراء، يتم تجهيز كلاً المتغيرين إلى سلسلة طولها صفر ولذا تكتب Debug كلمة منفردة على كل خط. وعندما ينتهي الإجراء يقوم أكسس بالتخلص من متغير Strnonstatic ولكنه يسترجع متغير Sstrstatic وقيمته.

الآن، قم بتشغيل الإجراء للمرة الثانية، في ثاني مرة يتم فيها تشغيل الإجراء يتم إعادة إنشاء وإعادة تجهيز متغير Strnonstatic إلى سلسلة طولها صفر ولكن يظل متغير Sstrstatic محتفظ بقيمته. ينتج عن العبارة المخصصة لمتغير Strnonstatic كلمة Nonstatic كما وضح فيما سبق بينما تلحق عبارة التخصيص لمتغير Sstrstatic كلمة Static للسلسلة الحالية. في كل مرة يتم فيها تشغيل الإجراء، يتم إعادة تجهيز المتغير غير الثابت وتقوم عبارة التخصيص بإعداد المتغير إلى كلمة Nonstatic ولكن يتم احتجاز قيمة المتغير الثابت وتقوم العبارة المخصصة بإلحاق كلمة Static إلى القيمة الحالية. يوضح الشكل "٨-٥ أ" إطار Immediate بعد تشغيل الإجراء ثلاث مرات.

يمكن جعل كل المتغيرات المحلية في إجراء ثابت عن طريق وضع الكلمة الأساسية Static في عبارة تعريف الإجراء، نستخدم المثال التالي:

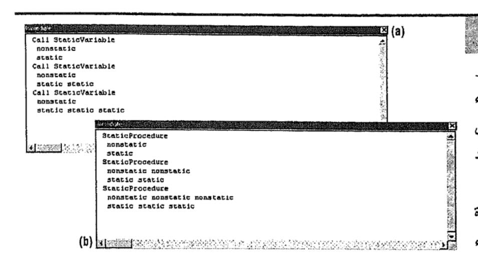
١- حدد كل العبارات في إجراء Staticvariable واضغط Ctrl+C لعمل نسخة. ضع نقطة الإدراج بعد الإجراء الأخير في الوحدة النمطية واضغط Ctrl+V لتبتم لصقها وتأكد من وجودك في طريقة عرض Full Module عن طريق تحديد Default في Full

Module View وذلك في مربع Options الذي يتم تحديده عن طريق Tools ⇦ Options. قم بتغيير عبارة تعريف الإجراء عن طريق إدراج الكلمة الأساسية Static وتغيير اسم الإجراء كما هو موضح فيما يلي:

```
Public Static Sub Staticprocedure()
    Dim Strnonstatic As String
    Static Sstrstatic As String
    Strnonstatic = Strnonstatic & "Nonstatic"
    Sstrstatic = Sstrstatic & "Static"
    Debug.Print Strnonstatic
    Debug.Print Sstrstatic
End
```

٢- انقر في إطار Immediate، اختر Edit ⇦ Select All واضغط Delete.

٣- قم بتشغيل إجراء Staticprocedure في إطار Immediate ثلاث مرات ولاحظ أن كلاً من المتغيرات أصبحت الآن ثابتة. يتجاوز استخدام الكلمة الأساسية Static في تعريف الإجراء عن الإجراء المستخدم فيه الكلمة الأساسية Dim داخل الإجراء. راجع الشكل "٨-٥ ب".

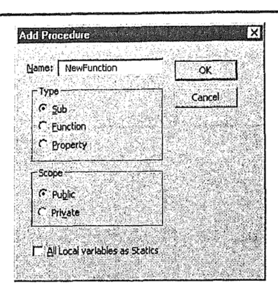


الشكل ٨-٥

يحتفظ المتغير الثابت بقيمته أثناء الاستدعاءات التي تتم للإجراء ويعاد تجهيز المتغير غير الثابت في كل مرة يتم استدعاء الإجراء "أ". يحدد استخدام الكلمة الأساسية Static في تعريف الإجراء مساحة التخزين لكل المتغيرات المحلية ويحتفظ بقيمهم طوال الوقت الذي تعمل فيه الوحدة النمطية إلى أن يعاد بدء أو يعاد تعيين الوحدة النمطية "ب".

٤- انقر في الوحدة النمطية basVariables وانقر زُر Reset في شريط الأدوات. عندما يعاد تعيين الوحدة النمطية، يتم التخلص من كل المتغيرات المُعلن عنها في الوحدة النمطية وبذلك تترك مساحة التخزين الخاصة بهم فارغة.

عند إدراج إجراء جديد، يمكن جعل كل المتغيرات المحلية ثابتة عن طريق تحديد Insert ⇦ Procedure واختيار All Local Variables as Statics في مربع Add Procedure، راجع الشكل "٨-٦". مع تحديد هذا الخيار يقوم برنامج VBA بإدراج الكلمة الأساسية Static في تعريف الإجراء بدلاً من المستخدم.



الشكل ٨-٦
حدد خيار ALL Local Variables As Statics في مربع Add Procedure لحفظ قيم المتغيرات المحلية المُعلن عنها في الإجراء.

تمرير البيانات إلى الإجراء

تتيح قابلية استدعاء إجراء من إجراء آخر فصل التعليمات البرمجية إلى إجراءات أصغر وأبسط. عندما يتم استدعاء إجراء من آخر، يمكن إرسال بيانات إلى الإجراء الذي يتم استدعائه ويمكن إرسال بيانات حرفية أو متغيرات.

تمرير بيانات حرفية

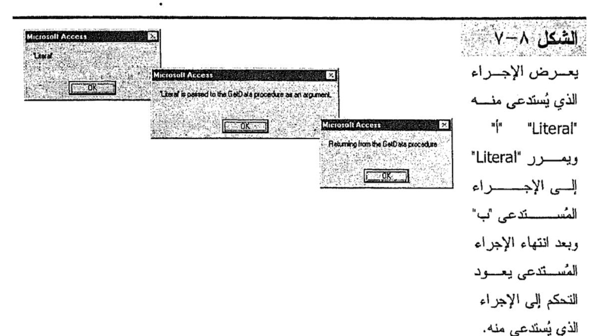
نستعرض تمرير البيانات الحرفية إلى الإجراء الذي تم استدعائه. في الوحدة النمطية Basvariables، أدرج الإجراءات الفرعية Passingliteral وGetdata:

```
Public Sub Passingliteral()
   Msgbox "Literal"
   Call Getdata("'Literal'")
   Msgbox "Returning From The Getdata Procedure"
End Sub
Public Sub Getdata (A As String)
   Msgbox A & " Is Passed To The Getdata Procedure As An Argument."
End Sub
```

تستدعي العبارة Call Getdata("'Literal'") في إجراء Passingliteral إجراء Getdata وتمرر القيمة الحرفية "Literal".

قم بتشغيل إجراء Passingliteral في إطار Immediate حيث يعرض إجراء Passingliteral رسالته، راجع الشكل "٨-٧أ" ثم يستدعي إجراء Getdata ويمرر له القيمة الحرفية. يستبدل VBA الحرف A في كل مكان يظهر فيه في الإجراء الذي تم استدعائه بالقيمة

التي تم تمريرها إليه. يعرض إجراء Getdata رسالته راجع الشكل "٨-٧ ب" ثم يختفي. يرجع VBA إلى إجراء Passingliteral في العبارة التالية ويعرض الرسالة، راجع الشكل "٨-٧ ج" ثم يختفي.



الشكل ٨-٧
يعرض الإجراء الذي يُستدعى منه "أ" "Literal" ويمرر "Literal" إلى الإجراء المُستدعى "ب" وبعد انتهاء الإجراء المُستدعى يعود التحكم إلى الإجراء الذي يُستدعى منه.

تمرير متغير إما بواسطة المرجع Reference أو بواسطة القيمة Value

يقدم برنامج أكسس طريقتين لإرسال بيانات متغير إما بإرسال المتغير نفسه أو بإرسال نسخة من قيمة المتغير.

عند إرسال المتغير نفسه يمكن للإجراء الذي يتم استدعائه أن يتعامل مع المتغير ويغير قيمته. يسمى إرسال المتغير نفسه "تمرير المتغير بواسطة المرجع" وهي الطريقة الافتراضية لإرسال متغير.

عند إرسال نسخة من قيمة المتغير يقوم VBA بإنشاء نسخة في موقع تخزين مؤقت آخر في الذاكرة ويُرسل النسخة. يمكن للإجراء الذي تم استدعائه استخدام النسخة ويمكن كذلك تغيير قيمتها ولكن لأن الإجراء المُستدعى يعمل باستخدام نسخة فلا يتأثر المتغير نفسه. ويسمى إرسال نسخة من المتغير "تمرير المتغير بواسطة القيمة".

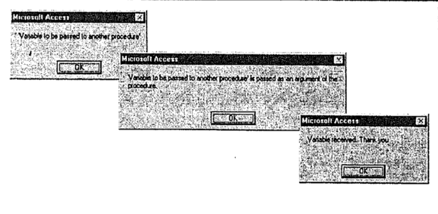
لتحديد أي الطرق تُستخدم، يجب أن تسبق اسم الوسيطة في قائمة الوسائط الخاصة بالإجراء الذي يتم استدعائه إما بالكلمة الأساسية **Byref** لتمرير المتغير نفسه أو بالكلمة الأساسية Byval لتمرير النسخة كما يلي:

```
[Byref|Byval] Argumentname [As Type]
```

لنستكشف ذلك، نقوم بإنشاء إجراءين لتمرير متغير أولاً بواسطة المرجع ثم بالقيمة. في الوحدة المنطية Basvariable. قم بإنشاء إجراءات Passingvariablebyref و Getvariablebyref كما يوضح ما يلي:

```
Public Sub Passingvariablebyref()
    Dim Strvariable As String
    Strvariable = "'Variable To Be Passed To Another Procedure'"
    Msgbox Strvariable
    Call Getvariablebyref(Strvariable)
    Msgbox Strvariable
End Sub
Public Sub Getvariablebyref (Byref Stra As String)
    Msgbox Stra & " Is Passed As An Argument Of The Procedure."
    Stra = "Variable Received. Thank You"
End Sub
```

يعرف إجراء Passingvariablebyref متغير Strvariable ويعين ثم يعرض قيمة السلسلة، راجع الشكل "٨-٨ أ". يستدعي الإجراء Getvariablebyref ويمرر له المتغير. يستقبل إجراء Getvariablebyref المتغير نفسه ويعرض المتغير، راجع الشكل "٨-٨ ب" ويغير قيمته. عند انتهاء إجراء Getvariablebyref. يرجع VBA إلى العبارة التالية من الإجراء الذي يتم الاستدعاء منه ويعرض المتغير القابل للتغيير وينتهي، راجع الشكل "٨-٨ ج".



الشكل ٨-٨

عند تمرير متغير "أ" بواسطة المرجع، يستقبل الإجراء المستدعى المتغير نفسه "ب" ويمكن تغيير قيمته. يستخدم الإجراء الذي يتم الاستدعاء بواسطته القيمة المتغيرة "ج".

اتبع هذه الخطوات لعرض كيفية عمل المتغيرات التي تم تمريرها:

١- قم بتشغيل إجراء Passingvariablebyref في إطار Immediate.

٢- في إطار Module، قم بعمل نسخة من كل التعليمات البرمجية لكل من الإجراءات إلى الحافظة. ضع نقطة الإدراج أسفل الإجراء الأخير وقم بلصق محتويات الحافظة Clipboard.

٣- حدد الإجراءات التي تم لصقها واختر Edit ⇦ Replace. ادخل Byref في مربع نص Findwhat وادخل Byval في مربع نص Replace With. تأكد من أن خيار Selected Text قد تم تحديده في مجموعة خيارات Search وانقر Replace All، راجع الشكل "٨-٩". انقر OK لتأكيد الاستبدالات الأربعة. حدد الإجراءات التي تم لصقها مرة أخرى، ادخل Passed في مربع نص Find What وادخل Passed By Value في مربع نص Replace With وانقر Replace All. انقر OK لتأكيد الاستبدالين واغلق مربع Replace.

٤- قم بتشغيل إجراء Passingvariablebyval في إطار Immediate.

الشكل ٨-٩

يُستخدم مربع Replace لضبط الإجراءات عن طريق استبدال القيم.

: Replace		
Find What:	ByRef	Find Next
Replace With:	ByVal	Cancel
Search: ( ) Current Procedure, ( ) Current Module, ( ) Current Database, (•) Selected Text	Direction: All; [ ] Find Whole Word Only; [ ] Match Case; [ ] Use Pattern Matching	Replace / Replace All

هذه المرة، يتم تمرير نسخة من قيمة المتغير إلى الإجراء المُستدعى أما الإجراء الذي يتم الاستدعاء منه فيقوم بعرض المتغير، راجع الشكل "٨-١٠ أ". يستقبل الإجراء المُستدعى نسخة من المتغير ويعرض رسالة راجع الشكل "٨-١٠ ب" ثم يغير القيمة كما حدث فيما سبق ولكن في تلك الحالة يتم التغيير في النسخة فقط. بعد الانتهاء من الإجراء المُستدعى، يرجع VBA إلى الإجراء الذي يتم الاستدعاء منه ويعرض مربع الرسالة الأخيرة بالقيمة في المتغير غير القابل للتبديل راجع الشكل "٨-١٠ ج".

على الرغم من اختلاف الاصطلاحات، تعتبر الطريقتين مناظرتين للطرق المعتادة في الربط وتضمين المستند. عند تمرير وسيطة Byref، فإنك تربط الإجراء بالمتغير وذلك ليتم تنفيذ الإجراءات التي تقع على الإجراء على المتغير نفسه. عند تمرير وسيطة Byval، فإنك تقوم

بتضمين نسخة من المتغير ولا يوجد هناك ارتباط بالمتغير نفسه وبذلك التغييرات التي تحدث في الإجراء لا تحدث في المتغير. إرسال متغير بواسطة المرجع أسرع لأن VBA لا يحتاج إلى الوقت المستخدم في إنشاء نسخة من المتغير.

الشكل ٨–١٠

عند تمرير متغير بواسطة القيمة "أ"، يستخدم الإجراء المُستدعى نسخة من المتغير "ب". على الرغم من أن الإجراء المُستدعى قد يغير قيمة النسخة، يحتفظ المتغير الأصلي بقيمته "ج".

ملاحظة يجعل تمرير المتغيرات بواسطة المرجع التفرقة بين إجراء دالة وإجراء فرعي غير واضح. عند تمرير متغيرات بواسطة مرجع يمكن للدالة أو الإجراء الفرعي الذي يستقبل المتغيرات التي تم تغييرها. هذا يعني أن كلاً من إجراءات الدالة والإجراءات الفرعية يمكن أن ترجع قيم من خلال الوسائط التي يتم تمريرها بواسطة المرجع. يكون لإجراء دالة فقط القدرة على إرجاع قيمة إضافية منفصلة عن القيم التي تم إرجاعها عن طريق الوسائط.

تمرير الوسائط إلى إجراء

توجد طريقتين لتمرير الوسائط إلى إجراء: عن طريق الترتيب أو عن طريق الاسم.

عند تمرير وسائط عن طريق الترتيب، يتم وضع قيم الوسائط في قائمة بالترتيب المحدد في عبارة الإجراء. إذا تم حذف وسيطة اختيارية، يجب تضمين فاصلة كحرف نائب لوسيطة مفقودة في قائمة الوسائط.

عند تمرير وسائط بالاسم، يجب تحديد اسم الوسيطة يتبعها علامة النقطتان وعلامة يساوي (:=) عامل التعيين يتبعهن قيمة الوسيطة. يمكن وضع الوسائط المسماة في أي ترتيب في القائمة ويمكن حذف أي وسائط اختيارية.

كمثال، يحدد تعريف إجراء Concatenate وسائطه كما يلي:

```
Public Function Concatenate(A,B)
```

تستدعي كلاً من العبارات التالية الدوال:

```
Call Concatenate("Johnson", "Diane")
Call Concatenate(B:="Diane", A:="Johnson")
```

تم إنشاء دالة Concatenated في الفصل السابق ولذا تم تخزينها في قاعدة بيانات أخرى. يمكن إتاحة الدالة في قاعدة البيانات الحالية عن طريق إعداد مرجع لقاعدة بيانات Ch7_Examples.

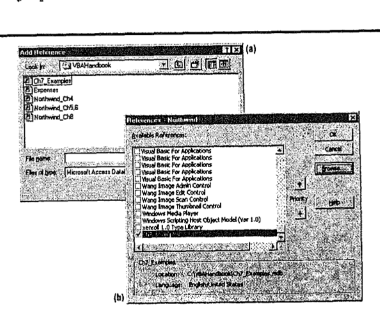
١- اختر Tools ⇦ Reference وانقر زر Browse في مربع References.

٢- في مربع Add Reference، اختر Microsoft Access Database في قائمة سرد وتحرير Files Of Type، حدد موقع قاعدة بيانات Ch7_Examples راجع الشكل "٨-١١ أ" وانقر OK. تتم إضافة المرجع إلى قاعدة البيانات إلى القائمة، راجع الشكل "٨-١١ ب".

٣- في إطار Immediate، ادخل كلاً من الآتي واضغط Enter.

```
? Concatenate("Johnson", "Diane")
? Concatenate(B:="Diane", A:="Johnson")
```

يوضح هذا المثال أنه عند تمرير وسائط بواسطة الاسم، يمكن ترتيب الوسائط في أي نظام.



الشكل ٨-١١
إضافة مرجع لقاعدة بيانات أخرى.

استخدام وسائط المتغير

يمكن إنشاء إجراء يقبل بيانات بها أكثر من نوع بيانات واحد عن طريق استخدام وسائط المتغير. على سبيل المثال، تضاعف دالة Multiply وسائطها وتقوم بإرجاع حاصل الضرب.

```
Public Function Multiply (X,Y)
    Multiply = X*Y
End Function
```

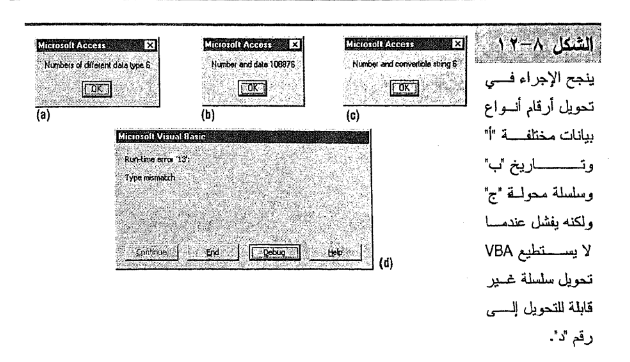
يمكن استدعاء هذه الدالة وتمرير قيمتين لها من أي نوع بيانات. إذا كان VBA قادر على تحويل أنواع البيانات لأنواع متوافقة فيمكن حساب حاصل الضرب.

يُعلن إجراء Callmultiply عن متغيرات لأنواع متعددة من البيانات ويحاول حساب ناتج الضرب عن طريق تمرير المتغيرات إلى دالة Multiply.

```
Public Sub Callmultiply
    Dim Intx As Integer, Sngy As Single
    Dim Dtmx As Date
    Dim Strx As String, Stry As String
    Intx = 2
    Sngy = 3
    Dtmx = #5/12/99#
    Strx = "2"
    Stry = "Three"
    Msgbox "Numbers Of Different Data Type " & Multiply(Intx, Sngy)
    Msgbox "Number And Date " & Multiply(Dtmx, Sngy)
    Msgbox "Number And Convertible String " & Multiply(Strx, Sngy)
    Msgbox "Number And Non-Convertible String " & Multiply(Intx, Stry)
End Sub
```

ادخل إجراء دالة Multiply العامة والإجراء العام الفرعي Callmultiply في الوحدة النمطية Basvariables ثم قم بتشغيل إجراء Callmultiply في إطار Immediate.

عند تمرير عددين من أنواع بيانات رقمية مختلفة ينجح VBA في تحويل أنواع البيانات، راجع الشكل "٨-١٢ أ" عند تمرير رقم وتاريخ يُحول VBA التاريخ لنظيره الرقمي ويحسب الناتج، راجع الشكل "٨-١٢ ب". عند تمرير رقم كسلسلة مثل "٢"، يُحول VBA السلسلة إلى نوع بيانات رقمي ويحسب الناتج، راجع الشكل "٨-١٢ج". وعند تمرير سلسلة مثل "Three". لا يستطيع VBA تحويل نوع البيانات ويعرض رسالة خطأ "Type Mismatch"، راجع الشكل "٨-١٢ د".



الشكل ٨-١٢ ينجح الإجراء في تحويل أرقام أنواع بيانات مختلفة "أ" وتاريخ "ب" وسلسلة محولة "ج" ولكنه يفشل عندما لا يستطيع VBA تحويل سلسلة غير قابلة للتحويل إلى رقم "د".

يمكن تجنب أخطاء وقت التشغيل في الإجراءات التي لها وسائط متغير بواسطة التحقق من نوع البيانات في الإجراء قبل أداء الحسابات. يمكن استخدام الدوال المضمنة الموضحة في الجدول ٨-١ لاختبار البيانات. راجع الفصل ٩ لمزيد من الأمثلة عن الإجراءات التي تختبر أنواع البيانات قبل إجراء الحسابات.

الجدول ٨-١: الدوال المضمنة لاختبار البيانات

الدالة	الوصف
Isnumeric(Expression)	ترجع الدالة True إذا أمكن التعرف على التعبير بأكمله كعدد وfalse في أي حالة أخرى. ترجع الدالة False إذا كان التعبير تاريخ. يمكن أن يكون التعبير أي سلسلة أو عدد.
Isdate(Expression)	ترجع الدالة True إذا أمكن تحويل التعبير إلى تاريخ وترجع False في أي حالة أخرى. يمكن أن يكون التعبير سلسلة أو تاريخ.
Isobject(Expression)	ترجع الدالة True إذا كان التعبير متغير لنوع Variant أو لنوع Object وترجع False في أي حالة أخرى.

الجدول ٨-١: الدوال المضمنة لاختبار البيانات

الدالة	الوصف
Isarray(Varname)	ترجع الدالة True إذا كان المتغير صف.
Isnull(Expression)	ترجع الدالة True إذا لم يحتوي التعبير على بيانات صحيحة. يمكن أن يكون التعبير أي سلسلة أو عدد.
Isempty(Expression)	إذا كان التعبير متغير تباين منفرد فإنه يرجع True إذا كان لم يتم تعيين المتغير كقيمة أو تم إعداده لقيمة Empty وترجع False في أي حالة أخرى.
Vartype(Varname)	إذا كان Varname اسم متغير به نوع بيانات أساسي فإنه يرجع قيمة تشير إلى نوع البيانات.
Typename(Varname)	إذا كان Varname اسم متغير به نوع بيانات أساسي، فإنه يرجع سلسلة تقدم معلومات بشأن المتغير.

تمرير كائنات كوسائط

يمكن تمرير كائنات كوسائط الإجراءات. كمثال، نقوم بإنشاء إجراء دالة لتغيير خاصية Caption لكائن. نستخدم نوع بيانات Object للوسيطة ليمكن تمرير أي نوع كائن له خاصية Caption، نستدعي بعد ذلك الدالة ونمرر لها الكائن الذي نريد تغيير عنوان التسمية له.

١- قم بإنشاء نموذج جديد يسمى Frmcaption. ضع عنصر تحكم تسمية على النموذج ثم اكتب Label على أساس أنها خاصية Caption وقم بإعداد خاصية Name على Ibicaption.

٢- ادرج دالة Changecaption كما يوضح ما يلي في الوحدة النمطية القياسية Basvariables.

```
Public Function Changecaption(Objectname As Object)
   Objectname.Caption = "My Caption"
End Function
```

٣- ضع زر أمر يسمى Cmdlabel على النموذج وقم بإعداد خاصية Caption له على Change Label Caption. انقر خاصية Onclick وانقر زر Build عن يمين مربع الخاصية. ادخل التعليمات البرمجية أسفله في قالب التعليمات البرمجية:

```
Private Sub Cmdlabel_Click()
    Call Changecaption (Lblcaption)
End Sub
```

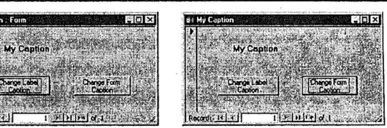
يستدعي هذا الإجراء دالة Changecaption وتمرر عنصر تحكم التسمية كوسيطة باستخدام العبارة القصيرة للإشارة إلى عنصر التحكم عند النقر على الزر يتغير عنوان التسمية إلى My Caption.

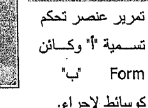
٤- ضع زر أمر يسمى Cmdform على النموذج وقم بإعداد خاصية Caption لــه على Change Form Caption. انقر في الوحدة النمطية للنموذج وادرج إجــراء الحدث أسفله، يعين VBA بصورة آلية الإجراء إلى زر خاصية Onclick.

```
Private Sub Cmdform_Click()
    Call Changecaption (Forms!Frmcaption)
End Sub
```

يستدعي هذا الإجراء دالة Changecaption ويمرر النموذج كوسيطة باستخدام عبارة نقطــة علامة التعجب للإشارة إلى النموذج بواسطة الاسم. عند النقر على الزر، يتغير عنــوان تسميــة النموذج إلى My Caption.

٥- احفظ النموذج وقم بالتبديل إلى طريقة عــرض Form. انقــر زر Change Label Caption، راجع الشكل "٨-١٣ أ". انقر زر Change Form Caption راجع الشكــل "٨-١٣ ب".





الشكل ٨-١٣
تمرير عنصر تحكم تســمية "أ" وكــائن Form "ب" كوسائط لإجراء.

تمرير النموذج كوسيطة

عند تشغيل إجراء تم تخزينه في وحدة نمطية لنموذج وترغب في استدعاء إجراء آخر وتمريــــر مرجع إلى النموذج يمكن استخدام خاصية Me للإشارة إلى النموذج، عند تشغيل إجراء في وحدة نمطية لنموذج تشير خاصية Me إلى النموذج. على سبيل المثال، انقر في نمــوذج Frmcaption وقم بالتبديل إلى طريقة عرض Design. انقر في الوحدة النمطية للنموذج وقم بتغيــــير إجــراء حدث Cmdform_Click() كما يوضح ما يلي:

```
Private Sub Cmdform_Click()
   Call Changecaption(Me)
End Sub
```

احفظ النموذج، قم بالتبديل إلى طريقة عرض Form وانقر زر Change Form Caption وبذلك يتغير عنوان تسمية النموذج كما سبق.

يمكن كذلك استخدام خاصية Form للإشارة إلى النموذج. على سبيل المثال، قم بتغيير إجراء حدث Cmdform_Click كما هو موضح واختبر الزر.

```
Private Sub Cmdform_Click()
   Call Changecaption(Form)
End Sub
```

استخدام خاصية Form لتمرير النموذج كوسيطة مفيد عند استخدام إجراء دالة حدث. على سبيل المثال، يمكن استدعاء دالة Change Caption لتغيير عنوان تسمية النموذج عند فتح النموذج. لرؤية كيفية عمل ذلك، انتقل إلى طريقة عرض Design، انقر خاصية Onclick في النموذج وادخل (Form) Changecaption= في خاصية حدث Onopen. احفظ النموذج وقم بالتبديل إلى طريقة عرض Form. يعمل إجراء الدالة ويتغير عنوان تسمية النموذج.

لا يمكن استخدام خاصية Me للإشارة إلى النموذج في إعداد خاصية الحدث. يسبب إدخال التعبير Changecaption(Me)= في خاصية حدث Onopen الخاصة بالنموذج غير قابل للفتح في طريقة عرض Form ويرجع النموذج إلى طريقة عرض Design.

استخدام عدد غير محدد من الوسائط

افتراضياً يتم طلب الوسائط في قائمة وسائط الإجراء. يحدث خطأ وقت التشغيل إذا فشلت في إرسال قيم لكل الوسائط. يكون من المناسب في بعض الأحيان إنشاء إجراء له عدد غير محدد من الوسائط وتوجد ثلاثة طرق لتقديم هذه الميزة التي تمتاز بالمرونة:

- تحديد أن الوسائط اختيارية.
- استخدام وسائط مع أنواع بيانات معرفة للمستخدم.
- استخدام وسائط الصفوف.

يوضح هذا الجزء كيفية تحديد وسائط اختيارية. راجع الإجراء الخاص "بإنشاء أنواع البيانات الخاصة بالمستخدم" و"استخدام الصفوف". فيما بعد في هذا الفصل للحصول على مزيداً من المعلومات بشأن تلك الطرق الخاصة بإنشاء إجراءات بعدد غير محدد من الوسائط.

يمكن تحديد أن بعض وسائط قائمة الوسائط اختيارية بواسطة سبق الوسيطة الاختيارية بالكلمة الأساسية Optional. يمكن للوسائط الاختيارية الحصول على أي نوع بيانات. يجب أن تكون الوسائط الاختيارية في نهاية قائمة الوسائط وبمجرد تحديد أن الوسيطة اختيارية، يجب تحدد كافة الوسائط التالية كوسائط اختيارية أيضاً.

لتوضيح تلك المفاهيم، لنفترض أنك تريد إنشاء دالة تحسب ناتج رقمين أو ثلاثة أرقام. يضاعف إجراء دالة Product ثلاثة أرقام عند تمرير ثلاثة متغيرات ولكنه يفشل عندما لا يتم إرسال وسيطة اختيارية وما يلي يوضح ذلك. في الوحدة النمطية Basvariables، أدرج إجراء Product.

```
Public Function Product (X, Y, Optional Z)
    Product = X*Y
    Product = Product*Z
End Function
```

في إطار Immediate، اكتب ?Product(2,3,4) واضغط Enter ثم اكتب ?Product(2,3) واضغط Enter. يتم حساب الناتج الأول ولكن الإدخال الثاني يسبب خطأ.

عندما تتسبب التعليمات البرمجية في إجراء في خطأ وقت التشغيل لأنـه لـم يتـم تمرير الوسيطة، يمكن استخدام دالة Ismissing لتحديد ما إذا كان قد تم تمرير وسيطة اختيارية. ترجع دالة Ismissing (Argname) القيمة True إذا لم يتم تمرير قيمة لوسيطة Argname وترجع القيمة False في أي حالة أخرى. في هذا المثال، يمكن تعديل دالة Product لتستخدم دالة Ismissing كما هو مبين فيما يلي.

```
Public Function Product (X, Y, Optional Z)
    Product = X*Y
    If Ismissing(Z) Then Exit Function
    Product = Product*Z
End Function
```

يحدد الإجراء المعدل ما إذا كانت الوسيطة الثالثة قد تم تمريرها إلى الدالة. إذا لم يتم تمرير الوسيطة الثالثة يخرج الإجراء بدون تنفيذ عبارة Product = Product*Z. راجع الفصل ٩ لمزيد من المعلومات بشأن اتخاذ قرارات في الإجراء.

تمرير بيانات لإجراء حدث

يحدد أكسس كلاً من الأسماء والوسائط لإجراءات الحدث الفرعية بصورة آلية. لا يمكن تغيير قائمة الوسائط بأي طريقة إلا كما ذكر فيما سبق. الهدف من قائمة الوسائط الافتراضية لحدث هو

تقديم وسيلة للاتصال بين تعليماتك البرمجية وأكسس. كمثال على ذلك، يكــون لإجــراء حــدث NOTINLIST المعرف بواسطة مربع تحرير وسرد العبارة التالية:

```
Private Sub Controlname_Notinlist(Newdata As String, Response As Integer)
```

تعتبر Newdata سلسلة قراءة فقط يستخدمها أكسس لتمرير النص الذي يدخله المستخدم فــي مربع النص جزء مربع التحرير والسرد، يمرر أكسس النص لإجراء الحدث. أمــا Response فهي ثابت يُستخدم لتحديد إما عرض أو إلغاء عرض الرسالة الافتراضية وإضافــة القيمــة فــي Newdata إلى قائمة مربع السرد والتحرير ويقوم إجراء الحدث بتمرير الثابت مرة أخرى إلــى أكسس.

ملاحظــة التغيير الوحيد الممكن إجرائه لقائمة الوسائط لإجراء حدث هــو اســتخدام البدايات القياسية لمتغيرات VBA. على سبيل المثال، يمكن استبدال الوسائط لإجراء حدث Notinlist مع Strnewdata وintresponse.

لا يكون لمعظم الأحداث وسائط مرتبطة، عل ســبيل المثــال، أحــداث مثــل Click وload وActivate وAfterupdate وInitialize وTerminate ليس لها وسائط. الحدث الذي تســتطيع إلغاء السلوك الافتراضي الذي يعقب الحدث له وسيطة تسمى Cancel لها نوع بيانات Integer، على سبيل المثال، لكل من الأحداث Dbiclick وunload وbeforeupdate وســيطة Cancel As Integer. ليتم إلغاء السلوك الافتراضي الذي يعقب الحدث، يتم إعداد وسيطة Cancel علــى True في عبارة تعيين في إجراء الحدث. توضح الفصــول القادمــة أمثلــة لتمريــر الوســائط لإجراءات حدث.

ملاحظــة للحصول على قائمة بالوسائط لإجراءات الحدث، راجــع جــدول "وســائط لإجــراءات الحــدث" علــى الاســطوانات المضغوطــة تحــت Tables\Chapter8.Pdf.

استخدام متغيرات مستوى الوحدة النمطية

يتم إنشاء متغير مستوى الوحدة النمطية بوضع عبارة تعريف فــي مقطــع Declarations فــي الوحدة النمطية باستخدام العبارة:

```
[Private|Public] Variablename As Type
```

تحدد الكلمة الأساسية التي تُستخدم في عبارة التعريف المتغير وهذا يعني أي الإجراءات تستطيع رؤية واستخدام المتغير. كما يحدث مع الإجراءات، يمكن لمتغيرات مستوى الوحدة النمطية أن تكون خاصة أو عامة. استخدم الكلمة الأساسية Private لإنشاء متغيرات وحدة نمطية خاصة والتي تستطيع الإجراءات داخل الوحدة النمطية فقط رؤيته. استخدم الكلمة الأساسية Public لإنشاء متغيرات وحدة نمطية عامة والممكن رؤيتها بواسطة كافة الإجراءات في الوحدات النمطية في قاعدة البيانات أو المشروع. يسمى متغير الوحدة النمطية العام متغير عمومي Global Variable ويمكن استخدام البادئة ٩ للإشارة إلى متغير وحدة نمطية عام.

اتبع تلك الخطوات لاستكشاف متغيرات مستوى الوحدة النمطية:

١- أعلن عن متغيرات مستوى الوحدة النمطية في مقطع Declarations لوحدة Basvariables النمطية كما هو موضح.

```
Private Strprivate As String
Public Gstrpublic As String
```

٢- ادرج Publicscoping وsamepublic الموضحة فيما يلي في الوحدة النمطية Basvariables. يعين إجراء Publicscoping قيمة "Public" لمتغير الوحدة النمطية العام Gstrpublic ويستدعي إجراء Samepublic المخزن في نفس الوحدة النمطية وإجراء Otherpublic المخزن في وحدة نمطية أخرى.

```
Public Sub Publicscoping()
   Gstrpublic = "'Public'"
   Call Samepublic
   Call Otherpublic
End Sub
Public Sub Samepublic()
   Msgbox Gstrpublic & " From The Same Module.", , "Samepublic"
End Sub
```

٣- قم بإنشاء وحدة نمطية جديدة تسمى Basother وادرج إجراء Otherpublic كما يلي:

```
Public Sub Otherpublic()
   Msgbox Gstrpublic & " From Another Module.", ,"Otherpublic"
End Sub
```

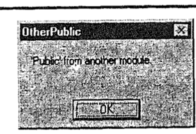
٤- اكتب Publicscoping في إطار Immediate واضغط Enter.

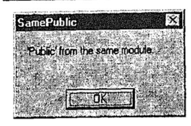
بعد تعيين قيمة للمتغير العام، يستدعي إجراء Publicscoping إجراء آخر في نفس الوحدة النمطية التي تعرض المتغير، راجع الشكل "٨-١٤ أ". ثم تستدعي إجراء آخر في وحدة نمطية

أخرى لعرض المتغير راجع الشكل "٨-١٤ ب" ويرى الإجراءين اللذين تم استدعائهما متغير مستوى الوحدة النمطية العام.

الشكل ٨-١٤

يمكن أن يُرى متغير مستوى الوحدة النمطية بواسطة إجراءات في نفس الوحدة النمطية "أ" وإجراءات في أي وحدة نمطية أخرى في المشروع "ب".





٥- ادرج إجراءات Privatescoping و sameprivate كما يوضح ما يلي في الوحدة النمطية Basvariables. يعين إجراء Privatescoping القيمة "Private" لمتغير مستوى الوحدة النمطية الخاص Strprivate ويستدعي إجراءات في نفس الوحدة النمطية وخارجها.

```
Public Sub Privatescoping()
    Strprivate = "'Private'"
    Call Sameprivate
    Call Otherprivate
End Sub
Public Sub Sameprivate()
    Msgbox Strprivate & " From The Same Module.", , "Sameprivate"
End Sub
```

٦- ادرج Privatescoping في الوحدة النمطية Basother:

```
Public Sub Otherprivate()
    Msgbox Strprivate & " From Another Module.", , "Otherprivate"
End Sub
```

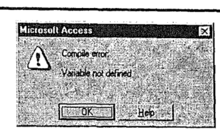
٧- اكتب Privatescoping في إطار Immediate واضغط Enter.

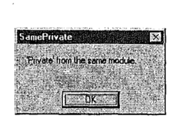
بعد تعيين قيمة للمتغير الخاص، يستدعي الإجراء Privatescoping إجراء آخر في نفس الوحدة النمطية والذي يعرض المتغير، راجع الشكل "٨-١٥ أ". يستدعي الإجراء Privatescoping إجراء آخر في وحدة نمطية أخرى. هذه المرة ينتج عن VBA خطأ وقت التشغيل، راجع الشكل "٨-١٥ ب" لأن متغير Strprivate خاص بالإجراءات في الوحدة النمطية Basvariables وهو غير مرئي بالنسبة لإجراء Otherprivate المخزن في الوحدة النمطية Basother.

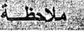
٨- انقر زر Reset.

الشكل ٨-١٥

يمكن أن يُرى متغير مستوى الوحدة النمطية الخاص فقط بواسطة الإجراءات في الوحدة النمطية نفسها "أ" وليس بواسطة الإجراءات في أي وحدة نمطية أخرى "ب".







لتعريف متغير خاص على مستوى الوحدة النمطية، يمكن استخدام الكلمة الأساسية Dim بدلاً من Private ولكن استخدام الكلمة الأساسية Private يجعل من الأسهل فهم التعليمات البرمجية.

يمكن كذلك استخدام متغيرات مستوى الوحدة النمطية العام المعرف في فئة وحدة نمطية مستقلة أو قياسية وليس في وحدة نمطية لنموذج أو تقرير في أي قواعد بيانات أخرى التي تشير إلى قاعدة البيانات التي يُعلن فيها عن المتغيرات العامة. يمكن قصر متغيرات مستوى الوحدة النمطية العام على قاعدة البيانات الحالية بواسطة تضمين عبارة Option Private Module في مقطع Declarations في فئة الوحدة النمطية القياسية أو المستقلة التي تم إعلانهم فيها. لا يختلط

الأمر عند استخدام الكلمة الأساسية Private، عند استخدامها في عبارة Option Private Module تعني الكلمة الأساسية Private خاص بقاعدة البيانات ولكن عند استخدامها في متغير أو في عبارة تعريف إجراء فإنها تعني خاص بالوحدة النمطية.

فهم مجال متغيرات مستوى الوحدة النمطية

تفرق قواعد "المجال" الوضوح بين متغيرات مستوى الوحدة النمطية التي تم إنشائها في الوحدات النمطية القياسية وفئات الوحدات النمطية المستقلة وبين متغيرات مستوى الوحدة النمطية التي تم إنشائها في الوحدات النمطية للنموذج والتقرير. الهدف الأساسي للوحدة النمطية للنموذج أو التقرير هي تخزين الثوابت والمتغيرات والإجراءات التي تطلب بواسطة نموذج أو تقرير. للتماشي مع هذا الهدف يكون مجال الثوابت والمتغيرات المُعلن عنها في وحدة نمطية لنموذج أو تقرير محدد للغاية. بالتحديد لا يمكن إنشاء ثوابت عامة على الإطلاق وفقط بعض المتغيرات يمكن أن تصبح عامة. ما يلي هو التفاصيل الدقيقة لمتغيرات مستوى الوحدة النمطية في وحدة نمطية لنموذج او تقرير:

- لا يمكن تعريف متغير عام كسلسلة طولها ثابت.
- لا يمكن تعريف صف عام.
- تكون المتغيرات العامة التي تم إنشائها وحدة نمطية لنموذج أو تقرير متاحة فقط لوحدات نمطية أخرى في قاعدة البيانات الحالية ولكنها ليست متاحة لقواعد بيانات أخرى.

يمكن للإجراءات التي تم تخزينها في مكتبات ارتباط حيوي "إجراءات DLL" والمُعلن عنها في مقطع Declarations في الوحدة النمطية أن تكون عامة أو خاصة. يمكن أن يُستدعى إجراء DLL المعرف مع عبارة Public Declare بواسطة أي إجراء في الوحدة النمطية وإجراء DLL المعرف مع Private Declare يمكن استدعائه فقط بواسطة الإجراءات في الوحدة النمطية القياسية أو المستقلة، يمكن تعريف كلاً من إجراءات DLL العامة والخاصة، في حالة عدم استخدام أي من الكلمتين الأساسيتين يكون إجراء DLL عام بصورة افتراضية. على عكس ذلك، في الوحدة النمطية للنموذج أو التقرير، يمكن تعريف فقط إجراءات DLL الخاصة.

استكشاف مدة حياة متغيرات مستوى الوحدة النمطية

عند فتح قاعدة بيانات أكسس لأول مرة، يفتح أكسس فقط الوحدات النمطية التي تحتاجها للبدء ثم تفتح وحدات نمطية كما تستدعي الإجراءات الموجودة في الوحدات النمطية. على سبيل المثال، إذا كان لنموذج وحدة نمطية لنموذج يتم تحميل هذه الوحدة النمطية على الذاكرة أول مرة يفتح فيها النموذج بينما يتم تحميل فئة وحدة نمطية أو وحدة نمطية قياسية أول مرة يتم استدعاء

إجراء مخزن فيها. عند تحميل وحدة نمطية في الذاكرة، يحدد VBA موقع لتخزين كل المتغيرات والثوابت المُعلن عنها في مقطع Declarations.

بمجرد تحميل وحدة نمطية في الذاكرة، لا يتم حذفها من الذاكرة أثناء تشغيل التطبيق ويعتبر هذا صحيحاً أيضاً بالنسبة للوحدات النمطية للتقرير أو النموذج ولا يحذف غلق النموذج أو التقرير الوحدات النمطية الخاصة به من الذاكرة. وبالتالي تستهلك متغيرات مستوى الوحدة النمطية ومتغيرات مستوى الإجراء الثابت الذاكرة وتحتفظ بقيمهم إلى وقت إغلاق قاعدة البيانات. تلميح الأداء الواضح هو استخدام متغيرات مستوى الوحدة النمطية فقط في حالة الضرورة القصوى.

اتبع الخطوات التالية لاستكشاف مدة حياة متغيرات مستوى الوحدة النمطية:

١- أعلن عن متغير مستوى وحدة نمطية عام في مقطع Declarations للوحدة النمطية Basvariables بواسطة كتابة Public Strmodule As String. تجعل الكلمة الأساسية Public المتغير متاح في إطار Immediate.

٢- ادرج إجراء Lifetimemodule الموضح فيما يلي في الوحدة النمطية Basvariables.

```
Public Sub Lifetimemodule()
    Strmodule = "'Module-Level Variable Exists Until You Close _
      The Database.'"
    Msgbox Strmodule, , "Lifetimemodule"
End Sub
```

يقوم إجراء Lifetimemodule بإعداد قيمة متغير Strmodule ويعرض رسالة ثم ينتهي.

٣- اكتب Lifetimemodule في إطار Immediate واضغط Enter. يعين ويعرض الإجراء القيمة، راجع الشكل "٨-١٦ أ".

يحتفظ المتغير Strmodule بالقيمة التي تم إعدادها بواسطة هذا الإجراء إلى ان يتم تغييرها أو إغلاق قاعدة البيانات أو إلى أن يتم النقر على زر Reset.

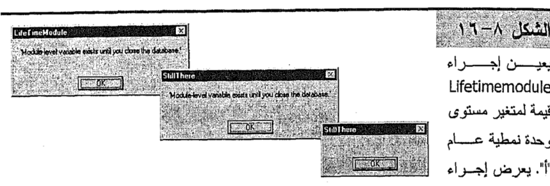
٤- ادرج إجراء Stillthere الموضح فيما يلي في الوحدة النمطية Basvariables. يعرض إجراء Stillthere القيمة الحالية لمتغير Strmodule.

```
Public Sub Stillthere()
    Msgbox Strmodule, , "Stillthere"
End Sub
```

٥- اكتب Stillthere في إطار Immediate واضغط Enter، يعرض مربع الرسالة القيمة الحالية، راجع الشكل "٨-١٦ ب".

٦- انقر زر Reset في شريط الأدوات. يتخلص VBA من كل المتغيرات العامة والخاصة في الوحدة النمطية ويترك مساحة التخزين الخاصة بهم خالية.

٧- قم بتشغيل إجراء Stillthere في إطار Immediate يظهر مربع الرسالة فارغ لأن المتغير تم إعادة تجهيز إلى سلسلة طولها صفر، راجع الشكل "٨-١٦ ج".



الشكل ٨-١٦
يعين إجراء Lifetimemodule قيمة لمتغير مستوى وحدة نمطية عام "أ". يعرض إجراء Stillthere القيمة الحالية للمتغير "ب". بعد إعادة تعيين الوحدة النمطية يتم إعادة تجهيز المتغير.

استخدام الثوابت

يمكن في حالة ظهور قيم ثابتة محددة بصورة متكررة جعل التعليمات البرمجية قابلة للقراءة بصورة أفضل باستخدام الثوابت. تعتبر الثوابت اسم ذو معنى يأخذ مكان رقم أو سلسلة لا تتغير. في معظم الأحيان، يقوم نظام التشغيل والتطبيقات المستخدمة مجموعات من الثوابت الفعلية ولكن يمكن إنشاء الثوابت المعرفة بواسطة المستخدم. يجعل استخدام الثوابت التعليمات البرمجية تعمل بصورة أسرع لأن VBA يكتب القيمة في الإصدار المجمع عند تجميع التعليمات البرمجية ولا ترغب في البحث عن القيم أثناء وقت التشغيل.

استخدام الثوابت الفعلية

يمكن في أكسس VBA استخدام الثوابت المعرفة بواسطة النظام والثوابت الفعلية في أكسس وكذلك ثوابت VBA الفعلية وأيضاً ثوابت JET الفعلية. الثوابت المعرفة بواسطة النظام هي ثوابت يقدمها نظام التشغيل أما الثوابت الفعلية فهي اسم ذو معنى يقدمه نظام التشغيل أما الثوابت الفعلية فهي اسم ذو معنى يقدمه التطبيق لاستبدال رقم لا يتغير.

استخدام الثوابت المعرفة بواسطة النظام

يوجد في أكسس أربعة ثوابت معرفة بواسطة النظام يمكن استخدامها في كافة كائنات قاعدة البيانات ماعدا في الوحدات النمطية وهم Yes وNo وOn وOff. كما يقدم أكسس ثلاث ثوابت إضافية معرفة بواسطة النظام يمكن استخدامها في كل كائنات قاعدة البيانات بما في ذلك الوحدات النمطية وهم: True وFalse وNull.

استخدام ثوابت أكسس الفعلية

يمتلك أكسس VBA مجموعة من الثوابت الفعلية الممكن استخدامها لتمثيل وسائط خصائص ودوال وطرق أكسس المتعددة ويكون لتلك الثوابت بادئ هو AC أمثلة لذلك هي: Accmdrefresh وAcform وAcprevious وAcpreview. من المزايا القيمة في ذلك النوع من الثوابت هو القدرة على تجميع الثوابت الفعلية. فعندما تطلب وسيطة لطريقة أو دالة أو خاصية ثابت فعلي يتم إعطاء مجموعة الثوابت للوسيطة والمسماة مجموعة الثوابت المعدودة للوسيطة اسم على سبيل المثال، وسيطة سجل طريقة Gotorecord لها مجموعة ثوابت فعلية تسمى Asrecord وتتضمن الثوابت المعدودة Acfirst وacgoto وaclast وacnewrec وacnext وacprevious.

استخدام ثوابت VBA الفعلية

يوجد في VBA كذلك مجموعة ثوابت فعلية تستخدم أساساً لتحديد إعدادات الخصائص ووسائط الطرق في تعليمات VBA البرمجية. ويكون لتلك الثوابت بادئ هو VB وعلى سبيل المثال Vbabort وvbquestion وvbcancel والممكن استخدامها لتحديد الوسائط لدالة Msgbox. يجمع VBA الثوابت الفعلية في فئات تتضمن Colorconstants وconstants وkeycodeconstants وكذلك Systemconstants. كما يجمع VBA الثوابت الفعلية في مجموعات ثوابت معدودة بما فيها البادئ VB في اسم المجموعة. على سبيل المثال، يعتبر Vbdayofweek هو اسم مجموعة الثوابت المعدودة Vbfriday وVbmonday وvbsaturday وvbsunday وvbthursday وvbtuesday وvbusesystemdayofweek وvbwednesday.

استخدام ثوابت JET الفعلية

يوجد في محرك قاعدة بيانات JET مجموعة ثوابت فعلية تستخدم أساساً لتحديد بيانات إعدادات خاصية دخول الكائن ووسائط الطريقة ويكون لهذه الثوابت بادئ هو DB وأمثلة لذلك هي

Dbdenyread وdbopentable وdbopendynaset والتـــي تســتخدم لتحديـــــد طريقـــة Openrecordset. يجمع JET أيضاً الثوابت الفعلية في مجموعات من الثوابت المعدودة ويطلـق أسماء على كل مجموعة باســـتخدام اســم وصفـــي والنهايـــة Enum. علـــى ســـبيل المثـــال، Recordsettypeenum هي مجموعة الثوابت المعدودة التي تمثل أنواع مجموعات الســـجلات بمـــا فـــي ذلـــــك Dbopendynamic وdbopendynaset وdbopenforwardonly وdbopensnapshot وdbopentable.

عرض الثوابت الفعلية

يمكن استخدام Object Browser لعرض الثوابت الفعلية لتطبيق. لفتح Object Browser مـن طريقة عرض Module انقر زر Object Browser في شريط الأدوات أو ضغط F2 أو اخـتر View ⇦ Object Browser.

في مربع حوار Object Browser، تتضمن قائمـــة الســـرد والتحريـــر Project/Library المشروع الحالي والمرجع الذي تم إعداده فيما سبق في مشروع Ch7_Examples، راجع الشكل "١٧-٨ أ". تتضمن أيضاً قائمة السرد والتحرير مراجع لمكتبات الكائنات الخمس التي تم إنشـائهم في دليل System في مجلد Windows عند تثبيت أكسس: أكسس (مايكروسوفت أكسس ٢٠٠٠) وdao (Microsoft Dao 3.5 Object Library For Jet) وjro (Microsoft Jet AND Replication Objects) وoffice (Microsoft Office 9.0 Library) وVBA (Visual Basic For Applications). تحتوي مكتبات الكائن السابق على مراجع للثوابت المقدمـــة مـن هـذه التطبيقات بالإضافة إلى معلومات مرجعية أخرى على الكائنات والأوامر. اتبع الخطوات التاليـــة لعرض بعض الثوابت الفعلية:

١- حدد أكسس كأنه مكتبة الكائن. حدد طريقة عرض Acform في مربع قائمـــة Classes في أقصى يسار مربع الحوار. يعرض مربـــع قائمـــة Members مجموعـــة الثوابـــت المعدودة، راجع الشكل "١٧-٨ ب". يعتبر Acformview هو اسم مجموعـــة الثوابـــت المعدودة لوسيطة طريقة العرض لطريقة Openform. يتضمن مربع قائمـــة Classes أسماء كل مجموعات الثوابت المعدودة للخصائص والدوال والطرق باستخدام البادئة AC للإشارة إلى اسم. يتم عرض الثوابت المعدودة في كل مجموعـــة فـــي مربـــع قائمـــة Members ولها البادئة AC.

٢- حدد Constants من مربع قائمة Classes في أقصى يسار مربع الحـــوار. يعـــرض مربع القائمة Members على جهة اليمين الثوابت الفعلية. اختر Acprompt في مربـــع قائمة Members وتعرض المساحة أسفل مربعات القائمة الرقم الـــذي يمـــاثل الثـــابت والمعلومات الإضافية بشأن الثابت. انقر زر Help إذا تم إتاحته، لعـــرض التعليمـــات الفورية.

٣- حدد DAO كمكتبة كائن. حدد Datatypeenum في مربع قائمة Classes. يعرض مربع قائمة Members مجموعة الثوابت المعدودة، راجع الشكل "٨-١٧ ج". يتضمن مربع قائمة Classes أسماء مجموعات الثوابت المعدودة باستخدام عبارة تتضمن اسم وصفي له نهاية Enum. على سبيل المثال، تعتبر Datatypeenum مجموعة الثوابت الفعلية التي تمثل أنواع البيانات المستخدمة بواسطة JET.

٤- اختر Dbboolean في مربع قائمة Members لعرض معلومات إضافية بشأن الثوابت.



قد تتغير القيم الرقمية الممثلة بواسطة الثوابت الفعلية في إصدارات حديثة من أكسس. لهذا السبب، يجب دائماً استخدام الثوابت الفعلية بدلاً من قيمهم الواقعية في تعليماتك البرمجية.

إنشاء الثوابت الخاصة بالمستخدم

يمكن إنشاء الثوابت الخاصة بك باستخدام عبارات التعريف ويتشابه إنشاء الثوابت مع إنشاء المتغيرات حيث يمكن إنشاء كل من ثوابت مستوى الوحدة النمطية والفرق هو أن عبارة التعريف لثابت تتضمن تعيين قيمة الثابت كما يلي:

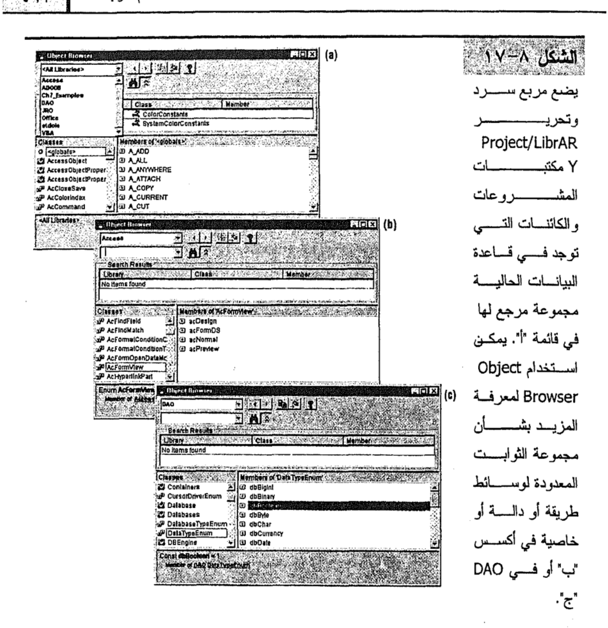
```
Const constantname As type = constantvalue
```

عندما يقرأ VBA عبارة التعريف لثابت فإنه ينشئ ويطلق اسم على موقع ذاكرة مؤقت ويحدد كمية التخزين المحددة بواسطة نوع البيانات الذي يحدده المستخدم ثم يخزن القيمة. يعتبر جزء As Type اختياري إذا لم يتم تحديد نوع بيانات يختار VBA نوع التخزين الأكثر ملائمة للقيمة التي أُدخلت.

إنشاء ثوابت مستوى الإجراء

يتم إنشاء ثابت لمستوى الإجراء وتعيين قيمته باستخدام عبارة تعريف داخل الإجراء كما يلي:

```
Const Constantname As Type = Constantvalue
```



الشكل ٨-١٧ يضع مربع سرد وتحرير Project/LibrAR Y مكتبات المشروعات والكائنات التي توجد في قاعدة البيانات الحالية مجموعة مرجع لها في قائمة "أ". يمكن استخدام Object Browser لمعرفة المزيد بشأن مجموعة الثوابت المعدودة لوسائط طريقة أو دالة أو خاصية في أكسس "ب" أو في DAO "ج".

على سبيل المثال، تقوم العبارة التالية بإعداد الثابت Intmax إلى قيمة عدد صحيح ١٤٤:

```
Const Intmax As Integer = 144
```

لا يظهر ثابت مستوى الإجراء خارج الإجراء.

إنشاء ثوابت مستوى الوحدة النمطية

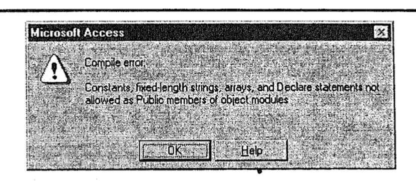
يتم إنشاء ثوابت مستوى الوحدة النمطية بواسطة وضع عبارة التعريف في مقطع Declarations في الوحدة النمطية. بالنسبة للوحدة النمطية القياسية، يمكن إنشاء ثوابت الوحدة النمطية الخاص والعام باستخدام العبارة:

```
[Public|Private] Const Constantname [As Type] = Constantvalue
```

في حالة حذف الكلمة الأساسية Public أو Private يكون الثابت خاص بصورة افتراضية ومتاح فقط بالنسبة للإجراءات في الوحدة النمطية التي يتم إنشاء الثابت فيها. على سبيل المثال، تقوم الأسطر التالية بإنشاء Gsnginterestrate كثابت عام يمكن رؤيته واستخدامه بواسطة أي إجراء في المشروع وsngtaxrate كثابت يمكن رؤيته واستخدامه بواسطة الإجراءات في الوحدة النمطية وهو غير مرئي للإجراءات في الوحدات النمطية الأخرى:

```
Public Const Gsnginterestrate As Single = 7.75
Private Const Sngtaxrate As Single = 32.5
```

بالنسبة للوحدة النمطية للنموذج والتقرير، يمكن إنشاء ثوابت خاصة فقط. وهذا يعني أن إدخال عبارة مثل Public Const Intmyconstant As Integer = 2 في مقطع Declarations للوحدة النمطية للنموذج أو التقرير يؤدي إلى خطأ، راجع الشكل "٨-١٨".



الشكل ٨-١٨
رسالة الخطأ التي تعرض عند محاولة تعريف ثابت عام في وحدة نمطية لنموذج أو تقرير.

استخدام الصفوف

تقوم VBA صفوف كطريقة للعمل بكفاءة مع متغيرات متعددة تكون لها نفس نوع البيانات. على سبيل المثال، يمكن إنشاء صف من عناصر تحكم مربع النص على النموذج أو صف من النماذج المفتوحة في قاعدة البيانات.

يعتبر الصفيف سلسلة من المتغيرات التي يشار إليها باستخدام نفس الاسم وباستخدام رقم يسمى "فهرس" لفصل المتغيرات. ويجب أن يكون للصف في المتغيرات نفس نوع البيانات. مع ذلك، إذا كان للصف نوع بيانات Variant يمكن أن تحتوي العناصر الفردية للصف على أنواع مختلفة من البيانات مثل الأرقام أو السلاسل أو الكائنات. يمكن إنشاء صف مع نوع أساسي أو نوع بيانات كائن أو نوع بيانات معرف بواسطة المستخدم. مجموعة كائنات Application وكائنات وصول البيانات التي تم مناقشتها في الفصول ٥ و ٦ هي صفوف كائنات. على سبيل المثال، تعتبر Forms صف للنماذج المفتوحة وtabledefs صف للجداول في قاعدة البيانات.

الهدف الأساسي من الصفوف هو جعل التعليمات البرمجية أبسط وأكثر كفاءة. عندما تريد معالجة مجموعة من العناصر في صف يمكن بسهولة الوصول إلى كل عنصر باستخدام رقم الفهرس لمتابعة التكرارات ويوضح الفصل ٩ التدخل في كل صف.

يمكن إنشاء صف به عدد ثابت من العناصر ويسمى صفيف الحجم الثابت الصفيف العادي. يمكن إنشاء صف بدون تحديد عدد العناصر ثم يتم ضبط حجم الصف بينما يعمل الإجراء ويعتبر الصف المعلن بدون عدد محدد من العناصر هو الصف الحيوي. لنفترض أنك تريد التحكم على النموذج يمكن استخدام صف ذو حجم ثابت لإمساك تلك القيم، مع ذلك إذا أردت استخدام الإجراء لأي نموذج يمكن استخدام الصف الحيوي لأن عدد عناصر التحكم قد يختلف بالنسبة لكل نموذج.

يمكن إنشاء صفوف محلية داخل الإجراءات والصفوف المشاركة في مقطع Declarations للوحدة النمطية. افتراضياً، يستخدم VBA صفر كأول رقم للفهرس (الحد الأسفل المنضم) للصف، في هذه الحالة يكون رقم الفهرس الأعلى (الحد العلوي المنضم) واحد أصغر من عدد العناصر. قد يكون الصفوف متعددة الأبعاد وقد يكون لها ما يصل إلى ٦٠ بُعد.

ملاحظة مطلب الذاكرة الأساسي لصف لأي نوع بيانات هو ٢٠ بايت بالإضافة إلى ٤ بايت لكل بُعد صف بالإضافة إلى عدد البايت للبيانات نفسها. تكون الذاكرة للبيانات نفسها هي ناتج عدد عناصر البيانات وحجم كل عنصر. يتطلب متغير Variant الذي يحتوي على صف على ١٢ بايت بالإضافة إلى الذاكرة المطلوبة من قبل الصف نفسه.

إنشاء صفوف الحجم الثابت

يُنشأ صف حجم ثابت لمستوى الإجراء باستخدام عبارة Dim أو Static داخل الإجراء أو في قائمة وسائط الإجراء. لتعريف الصف اتبع اسم الصف مع حدود لعدد الفهارس بين أقواس كما يوضح المثال التالي. يمكن استخدام البادئ في اسم المتغير للإشارة إلى صف إذا كنت تستخدم نمط تسمية Hungarian القياسي:

العبارة:

```
Dim astrNames (20) As String
```

تعرف صف يسمى Astrnames من العناصر ٢١ مع نوع بيانات String ومع أعداد الفهرس من صفر إلى ٢٠. عندما يكون الحد الأسفل المنضم للصف هو صفر يتم تضمين الحد المنضم العلوي في أقواس في عبارة التعريف.

العبارة:

```
Static Asngtaxrates (1 To 10) As Single
```

تعرف صف ثابت يسمى Asngtaxrates لعشر عناصر بها نوع بيانات Single وأرقام فهارس من واحد إلى عشرة. عندما يكون الحد المنضم الأسفل للصف أكبر من صفر، يتم تضمين كلاً الحدين والكلمة الأساسية TO بين الأقواس في عبارة التعريف.

العبارة:

```
Public Function Addresses (Astrnames(20) As String)
```

يعرف صف يسمى Astrnames يتكون من ٢١ عنصر بنوع بيانات String كوسيطة لدالة Addresses.

يمكن إنشاء صف ثابت الحجم لمستوى الوحدة النمطية عن طريق تعريف الصف في مقطع Declarations في الوحدة النمطية. استخدم الكلمة الأساسية Public ليشارك الصف بين كل الإجراءات في كل الوحدات النمطية في المشروع أو استخدم الكلمة الأساسية Private ليشارك الصف فقط بين الإجراءات في الوحدة النمطية التي يتم إنشاء الصف بها. كحالة استثنائية، لا يمكن إنشاء صف عام في وحدة نمطية لنموذج أو لتقرير.

ما يلي هو بعض الأمثلة لتعريفات الصفوف المشاركة. العبارة:

```
Public Aintmatrix (9,9) As Integer
```

تعرف صف عام له ذو بُعدين يسمى Aintmatrix لمائة عنصر مع نوع بيانات Integer له أزواج أرقام فهرس من (0,0) إلى (٩,9).

العبارة

```
Private Atxtamounts (1 To 6) As Textbox
```

تعرف صف خاص يسمى Atxtamounts يتكون من عناصر كائن مربع حوار مع أرقام فهرس من ١ إلى ٦.

إعادة تجهيز العناصر في الصف

يمكن استخدام عبارة Erase لإعادة تجهيز عناصر صف حجم ثابت. لاستكشاف الصفوف، قم بإنشاء وحدة نمطية جديدة تسمى Basarrays لتخزين الإجراءات التي يتم إنشائها في هذا المقطع ثم قم بإنشاء إجراء Createarray كما يوضح ما يلي:

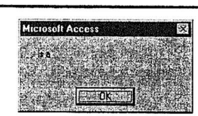
```
Public Sub Createarray()
    Dim Astrarray(3) As String
```

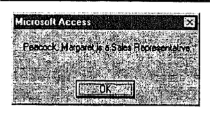
```
    Astrarray(0) = "Margaret"
    Astrarray(1) = "Peacock"
    Astrarray(2) = "Sales Representative"
    Msgbox Astrarray(1) & ", " & Astrarray(0) & " Is A " & Astrarray(2)
    Erase Astrarray
    Msgbox Astrarray(1) & ", " & Astrarray(0) & " Is A " & Astrarray(2)
End Sub
```

قم بتشغيل الإجراء في إطار Immediate. ينشئ الإجراء صف سلسلة حجم ثابت به ثلاثة عناصر ويعين قيم السلسلة ويعرض القيم في مربع الرسالة، راجع الشكل "٨-١٩ أ". بعد إبعاد الرسالة يستخدم الإجراء عبارة Erase لإعادة تجهيز العناصر إلى سلسلة طولها صفر ويعرض القيم التي تم إعادة تجهيزها في مربع الرسالة، راجع الشكل "٨-١٩ ب".

الشكل ٨-١٩

إنشاء صف حجم ثابت "أ" واستخدام عبارة Erase لإعادة تجهيز عناصر الصف "ب".





ملاحظة

تعمل عبارة Erase بشكل مختلف بالنسبة لصفوف الحجم الثابت والصفوف الحيوية. عند استخدام عبارة Erase لصف حيوي تكون مساحة التخزين المستخدمة بواسطة الصف الحيوي قابلة للاستعادة ومتاحة لإعادة الاستخدام. عموماً، عند استخدام عبارة Erase مع صف حجم ثابت، يتم إعادة تجهيز العناصر ولكن لا يتم استعادة مساحة الذاكرة. إذا أردت استخدام الصف لبعض الوقت فقط، يمكن تعريفه على أنه صف حيوي وبذلك تتم استعادة الذاكرة عند الانتهاء من استخدامه باستخدام عبارة Erase.

إنشاء صف بناء على قيم في قائمة

يمكن استخدام دالة Array لإنشاء صف بناء على قيم في قائمة باستخدام العبارة:

```
Array(Arglist)
```

تنشئ دالة Array صف باستخدام القيم في قائمة وسائط ARGLIST وترجع متغير VARIANT يحتوي على صف. يجب فصل القيم المتعددة بواسطة فواصل وفي حالة عدم وجود قيم في القائمة يتم إنشاء صف له حجم صفر.

ترجع الدوال Lbound وUbound أصغر وأكبر فهرس للبُعد المحدد للصف على التوالي:

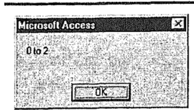
```
Lbound(Arrayname,Dimension)
Ubound(Arrayname,Dimension)
```

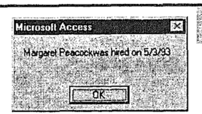
وسيطة Dimension الاختيارية هي الرقم بأكمله الذي يشير إلى أي حد منضم أعلى أو حد منضم أسفل للبُعد تريد إرجاعه حيث تكون ١ هي البُعد الأول و٢ هي البُعد الثاني وهكذا. وإذا حذفت البُعد يتم افتراض البُعد الأول.

لتقوم بعملية استكشاف باستخدام تلك الدوال، قم بإنشاء إجراء Arrayfunction الموضح فيما يلي في الوحدة النمطية Basarrays.

```
Public Sub Arrayfunction()
   Dim Vardata As Variant
   Vardata = Array("Margaret", "Peacock", #5/3/93#)
   Msgbox Lbound(Vardata) & " To " & Ubound(Vardata)
   Msgbox Vardata(0) & " " & Vardata(1) & " Was Hired On " & _
      Vardata(2)
End Sub
```

قم بتشغيل الإجراء في إطار Immediate. ينشئ الإجراء متغير متباين ويستخدم دالة Array لإنشاء صف له ٣ قيم في القائمة ويعين النتائج إلى المتغير المتباين. يجب أن يكون لمتغير Vardata نوع بيانات Variant من أجل تخزين القيم التي لها أنواع بيانات مختلفة. يعرض مربع رسالة الحدود المنضمة العلوية والسفلية للصف، راجع الشكل "٨-٢٠ أ" ويعرض مربع رسالة ثاني القيم، راجع الشكل "٨-٢٠ ب". في هذا الإجراء يحتوي متغير Vardata على صف.





الشكل ٨-٢٠

استخدم دوال Vbound وLbound لتحديد حجم الصف الذي تم إنشائه باستخدام دالة Array "أ". يعرض مربع الرسالة الثلاث قيم الخاصة بالصف "ب".

إنشاء صفوف حيوية

يستخدم صف حيوي بدلاً من صف حجم ثابت عندما لا تعلم مسبقاً عدد العناصر التي يتضمنها الصف. ينشأ الصف الحيوي باستخدام عبارتين: عبارة لتعريف الصف وعبارة لتحديد الحجم. يتم تعريف الصف باستخدام العبارة نفسها كما يحدث في صف الحجم الثابت بدون تحديد عدد العناصر، اترك فراغ بين الأقواس، واستخدم عبارة Redim في إجراء لتحديد الحجم وموقع مساحة التخزين المطلوبة بواسطة نوع البيانات المحدد.

إنشاء صف حيوي على مستوى الإجراء

يمكن إنشاء صف حيوي لمستوى الإجراء بتضمين عبارة تعريف داخل الإجراء باستخدام العبارة:

```
(Dim|Static) Arrayname() [As Type]
```

ثم تضمين عبارة Redim في الإجراء لتحديد الحجم. على سبيل المثال، تنشئ العبارة التالية صف حيوي محلي:

```
Dim Astrnames() As String
```

تعين هذه العبارة عدد العناصر إلى ٥ مع أرقام الفهرس من صفر إلى ٤:

```
Redim Astrnames (4)
```

إنشاء صف حيوي على مستوى الوحدة النمطية

يمكن إنشاء صف حيوي على مستوى الوحدة النمطية بوضع عبارة تعريف في مقطع Declarations في الوحدة النمطية باستخدام العبارة:

[Public|Private] Arrayname () [As Type]

ثم تضمين عبارة Redim في أي إجراء يشير إلى الصف لتحديد حجمه. على سبيل المثال، وضع العبارة التالية في مقطع Decalarations في الوحدة النمطية يُنشئ صف حيوي على مستوى الوحدة النمطية:

Public Asngtaxrates () As Single

وضع العبارة التالية داخل الإجراء يعين عدد العناصر إلى ٥ مع عدد الفهرس من ١ إلى ٥.

Redim Asngtaxrates (1 To 5)

إعداد الحدود المنضمة لصف حيوي

يمكن إعداد الحدود المنضمة لصف حيوي باستخدام متغيرات عدد صحيح كما في المثال التالي:

Redim Asngtaxrates (Intlower To Intupper)

تحدد عبارة Redim حجم الصف وتجهيز عناصر طبقاً لنوع بيانات الصف. هذا يعني أن Redim تجهز كل قيمة إلى صفر لصف رقمي وإلى سلسلة طولها صفر لصف سلسلة وإلى Empty لصف Variant وإلى Nothing لصف من الكائنات. يمكن استخدام عبارة Redim لتغيير حجم صف كلما رغبت في ذلك وعموماً تتخلص عبارة Redim من أي قيم مخزنة في الوقت الحالي في عناصر الصف عند تشغيل العبارة إلا إذا تم تضمين الكلمة الأساسية Preserve.

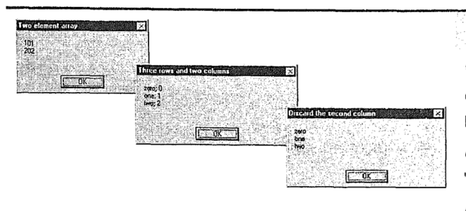
يمكن استخدام الكلمة الأساسية Preserve عندما ترغب في الاحتفاظ بالقيم الموجودة في الصف وكذلك تغيير حجم الصف في نفس الوقت. يمكن فقط إعادة ضبط حجم بُعد الصف الأخير. إذا خفضت حجم بُعد الصف الأخير، يتم التخلص من البيانات الموجودة في العناصر المستبعدة. عند استخدام الكلمة الأساسية Preserve لا يمكن تغيير عدد الأبعاد فقط حجم بُعد الصف الأخير.

لتوضيح تلك الأفكار، أنشئ إجراء Dynamicarray الموضح فيما يلي في الوحدة النمطية Basarrays:

```
Public Sub Dynamicarray()
    Dim Avar() As Variant, Str As String
    Redim Avar(2)
    Avar(0) = 101
    Avar(1) = 202
    Msgbox Avar(0) & Vbcrlf & Avar(1), ,"Two Element Array"
    Redim Avar(3,2)
    Avar(0,0) = "Zero": Avar(0,1) = 0
    Avar(1,0) = "One": Avar(1,1) = 1
    Avar(2,0) = "Two": Avar(2,1) = 2
    Str = Avar(0,0) & "; " & Avar(0,1) & Vbcrlf
    Str = Str & Avar(1,0) & "; " & Avar(1,1) & Vbcrlf
    Str = Str & Avar(2,0) & "; " & Avar(2,1)
    Msgbox Str, , "Three Rows And Two Columns"
    Redim Preserve Avar(3,1)
    Str = Avar(0,0) & Vbcrlf & Avar(1,0) & Vbcrlf & Avar(2,0)
    Msgbox Str, ,"Discard The Second Column"
End Sub
```

ينشئ الإجراء صف Avar كصف حيوي من المتغيرات ويستخدم ترتيب عبارة Redim لضبط حجم الصف. تضع أول عبارة Redim لضبط حجم الصف كصف مكون من عنصرين. تتلخص عبارة Redim الثانية من القيم المعنية وتعيد ضبط حجم الصف كصف ذو بُعدين به ٣ صفوف وعمودين. تتضمن كل من الثلاثة أسطر التالية عبارتي تعيين باستخدام الفاصلة المنقوطة لفصل العبارات. تتضمن عبارة Redim الثالثة الكلمة الأساسية Preserve لتحفظ ذلك مع القيم عندما يعاد ضبط حجم البُعد الأخير في الصف. يتم إعادة ضبط حجم الصف إلى صف مكون من ٣ عناصر به ٣ صفوف وعمود ويتخلص من بيانات العمود الثاني.

قم بتشغيل الإجراء في إطار Immediate. تتبع مربعات الرسائل التقدم الذي يحدث أثناء إنشاء الإجراء، راجع الشكل "٨-٢١".



الشكل ٨-٢١
يستخدم الإجراء الصف الحيوي وعبارة Redim لتمثيل صف مكون من عنصرين "أ" وصف مكون من بُعدين من ٣ صفوف و ٢ عمود "ب". يستخدم الإجراء عبارة Redim Preserve للتخلص من العمود الثاني "ج".

ملاحظة يمثل الثابت الفعلي Vbcrlf الخاص VBA حرف إرجاع يتبعها خط تغذية. استخدم الثابت لبدء خط جديد في مربع الرسالة.

استخدام الصفوف كوسائط

إذا لم ترد أن تحدد عدد الوسائط التي يستقبلها إجراء يتم استدعائه، يمكن استخدام الكلمة الأساسية Paramarray لأحد أرقام عشوائية من الوسائط من الإجراء الذي يستدعي ويضعهم في صف من المتغيرات. يجب أن تكون وسيطة Paramarray الوسيطة الأخيرة في قائمة الوسائط. على سبيل المثال، إذا كنت تنشئ إجراء عام لحساب متوسط مجموعة قيم رقمية، يمكن استخدام الكلمة الأساسية Paramarray في الوسيطة كما يلي:

```
Public Function Average(Paramarray Aargs())
```

وسيطة Aargs هي اسم الصف. عند استدعاء دالة Average يمكن إرسال عدد عشوائي من الوسائط كما توضح عبارة الاستدعاء التالية:

```
Call Average(2,5,46,23,1)
```

عند استخدام الكلمة الأساسية Paramarray تكون الوسائط هي نوع بيانات Variant بطريقة افتراضية ولا يمكن تحديد أي نوع بيانات آخر. لا يمكن تضمين الكلمات الأساسية Byref أو Byval أو Optional معاً مع الكلمة الأساسية Paramarray.

إنشاء أنواع البيانات الخاصة بالمستخدم

يمكن أيضاً إنشاء نوع البيانات الخاص بالمستخدم بناء على الأنواع الأساسية في جدول ٧-١ في الفصل ٧. يمكن إنشاء نوع بيانات مخصص لمتغير منفرد يجمع عناصر متعددة من المعلومات مع أنواع بيانات مختلفة. على سبيل المثال، يمكن استخدام متغير منفرد للإشارة إلى اسم عميل "نوع بيانات String" وتاريخ الطلب "نوع بيانات Data" وكمية الطلب "نوع بيانات Currency". يمكن تعريف نوع بيانات مخصص فقط في مقطع Declarations في الوحدة النمطية.

ينشأ نوع بيانات مخصص باستخدام عبارات Type وEnd Type. بعد تعريف نوع بيانات مخصص، يمكن تعريف متغير مع نوع البيانات المخصص كمتغير مستوى الإجراء أو متغير مستوى الوحدة النمطية. بمجرد تعريف متغير مع نوع البيانات المخصص يمكن الإشارة إلى عنصر من المتغير باستخدام العبارة:

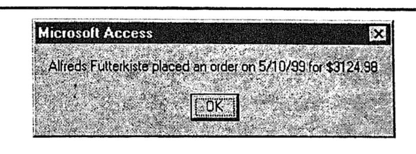
```
Variablename.Elementname
```

يمكن تعيين قيم لعناصرها باستخدام عبارات التعيين العادية.

لتوضيح تلك المفاهيم، انشئ نوع البيانات المخصص المسمى Orderinfo الموضح فيما يلي في مقطع Declarations لإجراء Basvariables وإجراء Vieworder الموضح في الوحدة النمطية Basvariable:

```
Public Type Orderinfo
Customer As String
Orderdate As Date
Amount As Currency
End Type
Public Sub Vieworder()
   Dim Ord As Orderinfo
   Ord.Customer = "Alfreds Futterkiste"
   Ord.Orderdate = #5/10/99#
   Ord.Amount = 3124.98
   Msgbox Ord.Customer &" Placed An Order On "& Ord.Orderdate & _
      " For $"& Ord.Amount
End Sub
```

قم بتشغيل إجراء Vieworder في إطار Immediate. يُعرف الإجراء متغير لنوع بيانات Orderinfo الجديد ويعين قيم لكل عنصر. يعرض مربع الرسالة عناصر متغير Ord، راجع الشكل "٨-٢٢".



الشكل ٨-٢٢
متغير منفرد مع نوع بيانات مخصص يحمل اسم العميل، تاريخ الطلب وكمية الطلب.

عند استخدام الكلمة الأساسية Public، تكون نوع البيانات المخصص متاح (مرئي) لكل إجراءات الوحدة النمطية في المشروع. استخدم الكلمة الأساسية Private بدلاً منها لتحديد مدى الرؤية للوحدة النمطية التي يتم تعريف نوع البيانات المخصص فيها. عند إنشاء نوع بيانات مخصص في وحدة نمطية قياسية يكون نوع البيانات المخصص عام بصورة افتراضية. عند إنشاء وحدة نمطية لنموذج أو تقرير يمكن لنوع بيانات مخصص أن يكون خاص فقط ولا يمكن تغيير مدى رؤيته مع الكلمة الأساسية Public.

يمكن أن تحتوي أنواع البيانات المعرفة بواسطة المستخدم على كائنات وصفوف ثابتة الحجم وصفوف حيوية كما توضح الأمثلة التالية:

```
Private Type Orders
Frminput As Form
Rptoutput As Report
Dborders As Database
End Type
Public Type Customerinfo
Customer As String
Address(2) As String
   'A Fixed-Size Array With Two Elements
Phone() As String
   'A Dynamic Array
Firstorder As Date
End Type
```

يمكن استخدام نوع بيانات مخصص منفرد لتمرير وسائط متعددة لإجراء باستخدام متغير منفرد. على سبيل المثال، يمكن تمرير المتغيرات الأربعة لنوع بيانات Customerinfo إلى إجراء Customerdataentry كمتغير منفرد كما يلي:

Public Sub Customerdataentry(Currentcustomer As Customerinfo)

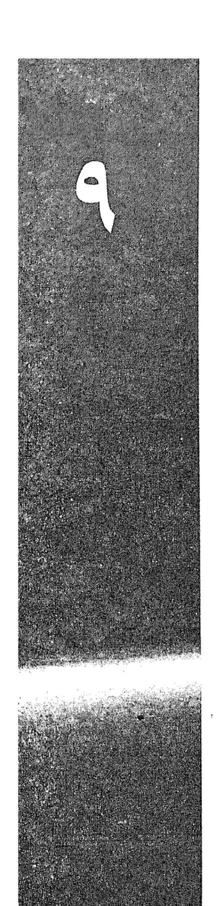
خلاصة

قدم هذا الفصل استخدام المتغيرات في الإجراءات. السبب الرئيسي لاستخدام المتغيرات هو أن تعمل التعليمات البرمجية بصورة أسرع عند استخدام متغيرات وتحديد أنواع البيانات وأنه من الممكن إنشاء إجراءات يمكن إعادة استخدامها لأنه من الممكن استخدام متغيرات لاستبدال أسماء كائنات محددة. أهم نقاط هذا الفصل هي:

- يمكن التحكم في أي الإجراءات الأخرى التي يمكنها استخدام متغير ومدة حياة هذا المتغير عن طريق تحديد المتغير على مستوى الإجراء أو على مستوى الوحدة النمطية وباستخدام الكلمات الأساسية في التعريف.
- يجب تعريف المتغيرات بأكثر أنواع البيانات تحديداً للحصول على تعليمات برمجية أسرع.
- يمكن تعريف متغيرات مستوى الإجراء في مكانين: في عبارة تعريف منفصلة داخل الإجراء أو في قائمة وسائط الإجراء الذي تم استدعائه بواسطة إجراء آخر.
- يمكن إنشاء متغير مستوى الإجراء بواسطة عبارة تعريف موضوعة داخل إجراء باستخدام عبارة Dim [Static] Variablename As Type أو كوسيطة في قائمة وسائط إجراء باستخدام عبارة Argumentname As Type.
- لا يكون متغير مستوى الإجراء مرئي لأي إجراء آخر بصورة مباشرة. الطريقة الوحيدة التي تجعل متغير مستوى الإجراء متاح لإجراء آخر هو تمرير متغير محلي كوسيطة للإجراء الآخر.
- عند انتهاء الإجراء، يتخلص أكسس من متغيرات مستوى الإجراء ويترك مساحات تخزينهم فارغة بصورة افتراضية. يمكن تجاوز هذا السلوك الافتراضي عن طريق تعريف متغير مستوى الإجراء بالكلمة الأساسية Static بدلاً من الكلمة الأساسية Dim أو باستخدام الكلمة الأساسية Static في تعريف الإجراء لجعل كل متغيراته المحلية ثابتة. يتم الاحتفاظ بقيم المتغير الثابت إلى أن يتم إعادة تشغيل أو إعادة تعيين الوحدة النمطية. لا يمكن إمداد فترة حياة وسيطة معلنة في قائمة وسائط إجراء.

- يمكن تعريف متغيرات مستوى الوحدة النمطية في مقطع Declarations للوحدة النمطية.
- يمكن تمرير قيم وكائنات لإجراء كوسائط بأسلوبين إما عن طريق المرجع حتى لا يستقبل الإجراء المتغير نفسه أو بواسطة القيمة حتى لا يستقبل الإجراء نسخة من الإجراء.
- يكون لإجراءات الحدث أسماء وقوائم وسائط سابقة التعريف.
- يمكن إنشاء ثوابت مخصصة لجعل التعليمات البرمجية قابلة للقراءة بصورة أفضل.
- يمكن استخدام صفوف لمعالجة متغيرات متعددة لها نفس نوع البيانات. إذا كان للصف نوع بيانات Variant قد يكون للعناصر أنواع بيانات مختلفة.
- يمكن إنشاء نوع بيانات مخصص لمتغير يستطيع التعامل مع عدة عناصر لها أنواع بيانات مختلفة.

يكمل الفصل التالي المقدمة إلى أكسس VBA بتوضيح العبارات الخاصة الممكن استخدامها للتحكم في أي التعليمات يتم تنفيذها وعدد مرات تنفيذ مجموعة من التعليمات.



التحكم في التنفيذ

يتم في هذا الفصل توضيح الإصدارات المختلفة التي يقدمها VBA من البنيات الخاصة بالتحكم. كما أنه يقدم العديد من تقنيات البرمجة مثل اختبار نوع البيانات الخاصة بالمتغيرات قبل إجراء الحسابات والطرق الجيدة التي تستخدم لتنفيذ حلقة عبر المجموعات فضلاً عن تنفيذ الحلقات التكرارية عبر مجموعة السجلات. كما يقدم الجزء الثاني من هذا الفصل بعض العبارات والوظائف المضمنة.

فهم البنيات الخاصة بالتحكم

يقوم VBA بتنفيذ العبارات الخاصة بالإجراء على نحو متتابع إلا إذا قمت بتحدي شئ آخر. يقوم VBA بتنفيذ كل عبارة من اليسار إلى اليمين ثم يقوم بالانتقال إلى العبارة التالية. وعندما يصل إلى عبارة End...statement ينتهي الإجراء. ويسمى هذا النوع من التنفيذ الذي يكون من اليسار إلى اليمين ومن أعلى إلى أسفل بالانسياب المتسلسل *sequential flow*.

يقدم VBA العديد من العبارات التي يمكن استخدامها لتغيير انسياب التنفيذ حيث يمكن استخدامها لتغيير التتابع الذي يحث به هذا التنفيذ والجمل التي تريد تكرارها والحمل التي تريد حذفها. وللقيام بذلك يجب استخدام كلمة أساسية أو عبارة تحكم لعمل البداية وكلمة أساسية أخرى لبيان النهاية. وتسمى عبارات التحكم التي تستخدم للتحكم في التعليمات البرمجية بنيات التحكم أو *control structure*. وتتأتى أغلب المميزات الخاصة ببرنامج VBA من نوعين من البنيات:

- تستخدم بنيات القرار *decision structure* لاختبار الشروط وتنفيذ العديد من العبارات بالاعتماد على ناتج الاختبار.
- تستخدم بنية الحلقة *loop structure* لتنفيذ مجموعة من العبارات البرمجية بشكل متكرر.

ملاحظة

قم بإنشاء نسخة جديدة من قاعدة البيانات Northwind التي تسمى Northwind_Ch9 ثم قم بإنشاء وحدة جديدة تسمى basControlStructure إذا كنت تريد اكتساب المزيد من الخبرة العملية. قم بإنشاء عينة من الإجراءات في وحدة basControlStructure أو في وحدة النموذج المحددة في المثال.

اتخاذ الإجراءات وفقاً للشروط

تستخدم بنية القرار لتنفيذ مجموعات مختلفة من العبارات البرمجية بناءاً على نتائج اختبار إحدى الشروط. ويمكن أن يكون هذه الشرط أي تعبير رقمي أو تسلسلي يتم تقييمه بالقيم True أو

False. ودائماً ما تكون هذه الشروط هي عبارة عن مقارنة مثل intCount > 0 كما أنها يمكن أن تكون أي تعبير يؤدي إلى وضعها في قيم شرطية مثل intCount. وإذا كانت القيمة الرقمية هي الصفر فإن الشرط يصبح False. أما إذا كانت القيمة الرقمية غير الصفر فإن الشرط يصبح True.

استخدام بنيات If...Then

افترض أن لك مجموعة من العبارات البرمجية التي تريد تنفيذها عندما يكون أحد الشروط True وإذا كان غير ذلك فإنك لا تريد تنفيذها. يمكنك في هذه الحالة استخدام بنية القرار If...Then، وهو يستخدم لتقييم شرط الاختبار وتنفيذ إحدى العبارات أو مجموعة منها إذا كانت قيمة اختبار الشرط هي True. ويمكن أن يكون اختبار الشرط أي تعبير يتم تقييمه بالقيم True أو False. ويوجد هناك نوعان من هذه البنية وهما صيغة السطر الواحد وصيغة السطور المتعددة.

ويتم كتابة صيغة السطر الواحد على هذا النحو:

```
If condition Then statements
```

إذا كانت قيمة شرط الاختبار هي True فإن العبارة الموجودة في هذا السطر يتم تنفيذها ثم ينتقل VBA إلى العبارة التالية في الإجراء. أما إذا كانت حالة الاختبار هي False فإنه لا يتم تنفيذها ثم يتم تنفيذ ما يليها.

ويتم كتابة صيغة السطور المتعددة على هذا النحو:

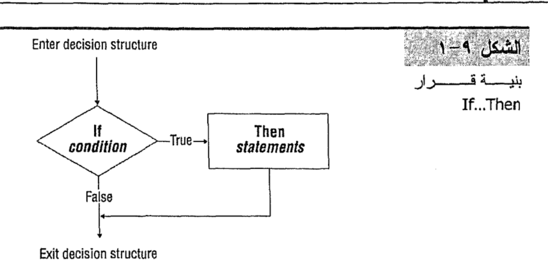
```
If condition Then
   statements
End If
```

وإذا كانت قيمة اختبار الشرط هي True فإن العبارة يتم تنفيذها ثم ينتقل VBA إلى العبارة End If ثم يخرج من بنية التحكم. أما إذا كانت قيمة اختبار الشرط هي False فإن ذلك يؤدي إلى التغاضي عن هذه العبارة، ثم ينتقل VBA إلى العبارة End If ثم الخروج من بنية التحكم، ويوضح ذلك الشكل ٩-١.

يمكن استخدام صيغ السطر الواحد لعدة عبارات إلا أن العبارات يلزم أن تكون في سطر واحد ويفصل بينها (:). وتفيد الصيغة ذات السطر الواحد عند إجراء الاختبارات البسيطة، إلا أن الصيغة التي تحتوي على العديد من الأسطر تكون في غاية الفائدة وأسهل في القراءة عند وجود عبارة واحدة.

الشكل ٩–١

بنيـــة قـــــرار If...Then



قم بإدخال إجراء NullToZero الموضح فيما يلي في الوحدة basControlStructure مـن قاعدة بيانات Northwind_Ch9 التي قد قمت بإنشائها للأمثلة الموضحة في هذا الفصل:

```
Public Function NullToZero(X)
   If IsNull(X) Then NullToZero = 0
End Function
```

وتقوم هذه الإجراءات باختبار وسيطة الوظيفة. وإذا كانت القيمة هي Null فإن الإجراء يقـوم بضبط القيمة على الصفر وإن لم يكن فإن هذا الأجراء لا يقوم بأي شئ. ومن الإطـــار الحـــالي اكتب NullToZero(4)? ثم اضغط على Enter. وبما أن الشرط هو False فإن الإجراء ينتـهي بدون إرجاع القيمة. والآن اكتب NullToZero(Null)? ثم اضغط على Enter. ويكون الشـــرط True وقيمة الوظيفة قد تم ضبطها على الصفر. فإن الإطار الحالي يعرض القيمة المسترجعة.

افتح النموذج Customers من وضع Design ثم انقر خاصية الحدث OnLoad الخاصـــة بالنموذج. ثم انقر زر Build. ادخل إجراء الحدث Form_Unload كما هو موضح فيما يلي:

```
Private Sub Form_Unload (Cancel As Integer)
   If MsgBox ("Close form? ", vbYesNo) = vbNo Then
      Cancel = True
   End If
End Sub
```

احفظ النموذج ثم انتقل إلى وضع Form ثم انقر مربع Close الخاص بـالنموذج. ثم يعرض الإجراء مربع رسالة Yes أو No. وإذا قمت بنقر زر No فإن الشرط يصبـــح True كما يقوم الإجراء بضبط وسيطةCancel على True. أما إذا نقرت زر Yes فإن الشرط تكـــون قيمته False وينتهي الإجراء ويقوم أكسس بإغلاق النموذج.

ملاحظـــة تتطلب بنيات التحكم الكلمات الأساســـية If وthen وIf end. وإذا قمـت بحذف أي كلمة أساسية في أي بنية تحكم فإن VBA يقوم بعمل خطأ تجميع.

استخدام بنية If...Then...Else

بفرض أنه هناك مجموعتان من العبارات البرمجية وهنك رغبة في تنفيذ مجموعة واحـــدة فقـط حسب قيمة الشرط فإنك في هذه الحالة سوف تستخدم بنية If...Then...Else التي تقيـــم شــرط اختبار مع تنفيذ مجموعة واحدة من العبارات إذا كانت قيمة الاختبار هي True وتنفذ المجموعــة الأخرى إذا كانت القيمة هي False. وكما هو الحال مع بنية If...Then فإنه يوجد صيـــغ ذات سطر واحد وصيغ أخرى لها عدة أسطر.

ويتم كتابة صيغة السطر الواحد على هذا النحو:

```
If condition Then statements [Else elsestatements]
```

ويقوم VBA بتقييم الشرط فإذا كان True فإنه ينفذ العبارة التي تلي Then وإلا فإنه العبارة الــتي تتبع Else. وبعد تنفيذ أياً منهما فإن VBA ينتقل إلى البارة التالية في الإجراء. ويمكن اســـتخدام صيغة السطر الواحد للعديد من العبارات إلا أن العبارات لابد أن تكون في سطر واحد مع فصلها بالعلامة (:). إلا أنه دائماً ما تستخدم صيغة السطر الواحد عند كتابة العبارات التي تحتوي علــى عبارة واحدة لكل بديل.

يمكن أن تستخدم صيغة السطر الواحد عندما يكون هناك اختبار شرط وزوج من المجموعات البديلة من العبارات البرمجية:

```
If condition Then
   [statements]
[Else
   elsestatements]
End If
```

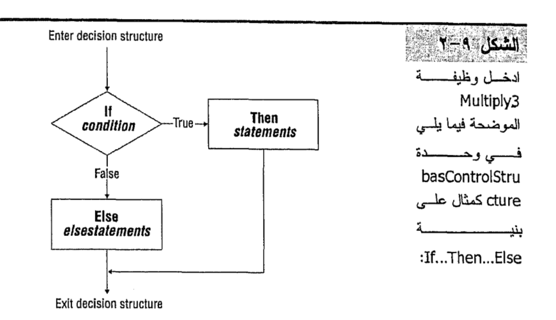
كما يمكن أن تستخدم هذه الصيغة أيضاً عندما يوجد أكثر من اختبار شرط واحد والعديد مــن العبارات المراد تنفيذها.

يمكن استخدام بنية If...Then...Else لتحديد أي العبارات المراد تنفيذها من بيــن مجموعــة كبيرة من العبارات كما يلي:

```
If condition1 Then
   [block1statements]
```

```
[ElseIf condition2 Then
    block2 statements]
...
[Else
    blockN statements]
End If
```

يقوم VBA بتقييم الشرط الأول وإذا كان True فإنه ينفذ الجزء الأول من العبارات ثم ينتقل إلى العبارة التالية التي تلي End If مع عدم تنفيذ ما بينهما. أما إذا كانت القيمة False فإن VBA يختبر الشرط التالي في عبارة ElseIf. وإذا كان الشرط True فإن VBA يقوم بتنفيذ مجموعة العبارات البرمجية. أما إذا لم يكن أي من الشروط True فإن VBA ينفذ مجموعة العبارات الاختيارية Else ثم يترك بنية القرار. والأسطر المطلوبة فقط هي If...Then وEnd If. يوضح الشكل ٩-٢ بنية القرار If...Then...Else في حالة الشرط الواحد.



الشكل ٩-٢

ادخل وظيفة Multiply3 الموضحة فيما يلي في وحدة basControlStructure كمثال على بنية If...Then...Else:

```
Public Function Multiply3(X,Y,Optional Z)
    If IsMissing(Z) Then Multiply3 = X*Y Else Multiply3 = X*Y*Z
End Function
```

تحدد وظيفة Multiply3 إذا كانت الوسيطة الاختيارية قد تم تمريرها كما أنها ستستخدم صيغ مختلفة تبعاً للنتائج. وفي الإطار الحالي اكتب Multiply3(2,3)? ثم اضغط على Enter. وفي هذه الحالة يكون الشرط False لذلك يقوم أكسس بضرب الأرقام الثلاثة مع عرض الناتج.

ثم ادخل وظيفة MultiplyTest الموضحة فيما يلي في وحدة basControlStructure.

```
Public Function MultiplyTest(X,Y)
   If IsNumeric(X) and IsNumeric(Y) Then
      MultiplyTest = X*Y
   Else
      MsgBox "Non-numeric arguments"
   End If
End Function
```

تحدد الوظيفة MultiplyTest إذا ما كانت الوسيطة تحتوي على بيانات رقمية من عدمه قبل إجراء عملية الضرب. وفي الإطار الحالي اكتب (MultiplyTest(2,3 ثم اضغط على Enter. إذا كان الشرط True فإن الأرقام يتم ضربها كما يتم عرض الناتج. والآن اكتب (MultiplyTest("two",3 ثم اضغط على Enter. وإذا كان False يتم طباعة الرسالة.

استخدام TypeOf...Is لتحديد نوع الكائن

دائما ما تحتاج إلى تحديد نوع عنصر التحكم عند العمل معها في النموذج. فمثلاً افترض أنك تريد استخدام نموذج للمراجعة والتحرير. فإنك بحاجة إلى التبديل إلى خاصية إغلاق عنصر التحكم بين نماذج عرض التحرير والمراجعة. إلا أن المشكلة هي أن ليس كل عناصر التحكم لها خاصية الإغلاق لذلك فإنه يجب عليك معرفة ما إذا كان أحد عناصر التحكم له خاصية إغلاق قبل ضبط الخاصية. ويمكن أن تستخدم التعبير TypeOf objectname Is objecttype كاختبار للشرط لتحديد ذلك.

يوضح الإجراء PreviousControlType إذا ما كان أحد عناصر التحكم هو مربع نصي كما أنه يقوم بعرض رسالة في هذه الحالة. ويتم إعداد هذا الإجراء كإجراء عام حيث يمكنه العمل مع أي نموذج عن طريق استخدام خاصية PreviousControl الخاصة بكائن Screen للإشارة إلى عنصر التحكم.

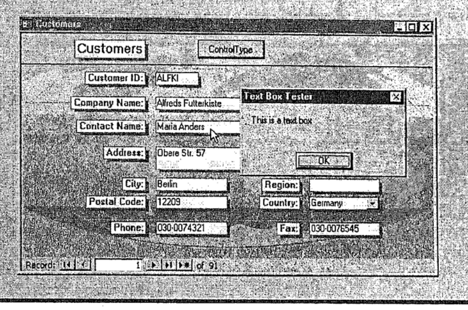
```
Public Function PreviousControlType()
   If TypeOf Screen.PreviousControl Is TextBox Then
```

Continued on next page

```
      MsgBox "This is a text box", , "Text Box Tester"
   End If
End Function
```

إذا كنت تريد اختبار هذه الوظيفة ادخلها في وحدة basControlStructure. ثم افتح نموذج Customers من وضع Design ثم قم بإضافة زر الأمر المسمى cmdControlType الذي يحتوي على العنوان Control Type. انقر خاصية OnClick ثم قم بضبط الخاصية على ()PreviousControlType=. ويمكنك عن طريق استخدام إحدى الوظائف المخزنة في الوحدة القياسية من إجراء الحدث لصق الزر في أي نموذج.

انتقل إلى وضع Form لاختبار أحد عناصر التحكم ثم انقر عنصر التحكم ثم الزر. وإذا قمت بنقر مربع النص يتم عرض مربع الرسالة وإلا لن تحدث أية استجابة. إذا نقرت إحدى التسميات فإن التركيز يتحول إلى عنصر تحكم مربع الحوار المرتبط بهذه التسمية ثم يؤدي نقر الزر إلى عرض هذه الرسالة.



استخدام بنية Select Case

إذا كان هناك مجموعات عدة من البدائل وكنت تريد تنفيذ واحدة منها فقط وفقاً للقيمة الخاصة بأحد التعبيرات فإنه يمكنك في هذه الحالة استخدام بنية القرار Select Case...End Select كبديل للبنية If...Then...Else. ومن بنية القرار Select Case يمكنك تقييم تعبير اختبار والتعامل مع حالات معينة عن طريق مقارنة القيمة الخاصة بتعبير اختبار معين بالنسبة للقيم المناسبة لكل حالة.

The Select Case decision structure has the syntax:

```
Select Case testexpression
   Case expressionlist1
```

```
      [block1 statements]
   Case expressionlist2
      [block2 statements]
...
   Case Else
      [blockN statements]
End Select
```

وتعتبر كل *expressionlist* قائمة تحتوي على قيمة واحدة أو أكثر من القيم الملائمة للحالة. وتحتوي العبارة المناظرة الملائمة على العبارات التي يتم تنفيذها إذا كانت قيمة تعبير الاختبار تتناسب مع *expressionlist*. ويمكن أن يكون للعنصر في *expressionlist* أياً من النماذج التالية:

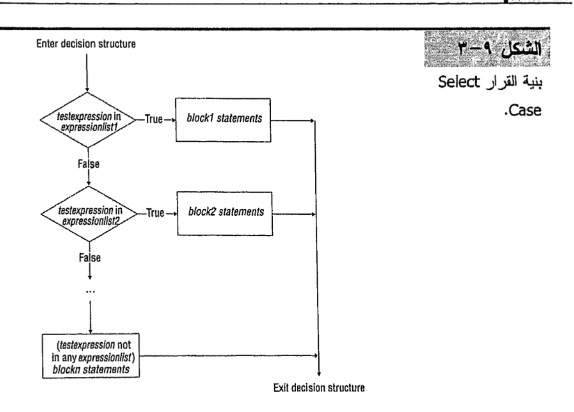
Expression	ويتضح المثال في Case 2,4,5,7
Expression To *expression*	ويجب أن يكون التعبير الذي يسبق To أصغر من التعبير الذي يتبعها. ويتضح المثال في Case 2 To 5
Is *operator expression*	ويمكن أن يكون العامل أي عامل مقارنة ما عدا Is أو Like. ويتضح المثال في Is>10

يختبر VBA قيم *expressionlist 1* بعد تقييم تعبير الاختبار في عبارة Select Case وإذا توافقت أي قيمة في *expressionlist 1* مع تعبير الاختبار فإن VBA يقوم بتنفيذ المجموعة ١ من عبارات البرمجة ثم ينتقل إلى العبارة التي تلي عبارة End Select مع التغاضي عما بينهما. أما إذا لم تتوافق أياً من القيم مع القيم الموجودة في تعبير الاختبار فإن VBA ينتقل إلى عبارة Case التالية مع اختبار القيم الموجودة في *expressionlist*. وإذا توافقت معها فإن VBA يقوم بتنفيذ العبارات البرمجية الموجودة في Case Else الاختيارية. أما إذا توافقت إحدى قيم تعبيرات الاختبار مع إحدى القيم الموجودة في أكثر من Case واحدة، يقوم VBA بتنفيذ العبارات التي تتوافق مع أول عبارات Case. يوضح الشكل ٩-٣ بنية القرار Select Case.

ملاحظة يمكن استخدام جملة TypeOf objectname Is objecttype في بنية التحكم Select Case.

الشكل ٩-٣

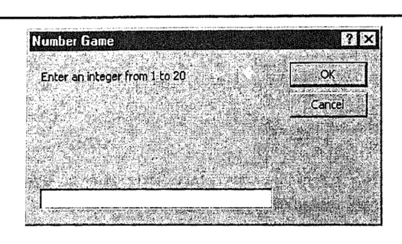
بنية القرار Select
.Case



كما هو موضح في المثال ادخل الإجراء NumberGame الموضح فيما يلي في وحدة .basControlStructure

```
Public Sub NumberGame()
   Dim intnumber as Integer
   intnumber = InputBox("Enter an integer from 1 to 20","Number
Game")
   Select Case intnumber
      Case 2, 4, 6, 8
         MsgBox "The number is even and less than 9"
      Case 9 To 20
         MsgBox "The number is from 9 to 20"
      Case Else
         MsgBox "The number is odd and less than 9"
   End Select
End Sub
```

يستخدم الإجراء وظيفة InputBox لتجميع العدد الصحيح من اللاعب "انظر الشكل ٩-٤".
يتم تعيين الرقم الذي تقوم بإدخاله إلى المتغير intnumber. تختبر بنية القرار Select Case قيمة الرقم في مقابل التعبيرات التي توجد في كل Case مع عرض مربع رسالة يخص case الذي تتوافق معه.



الشكل ٩-٤
يتم تعيين القيمة التي تقوم بإدخالها في مربع الإدخال إلى المتغير الذي يتم اختباره باستخدام بنية القرار Select Case.

قم بتشغيل الإجراء NumberGame في الإطار الحالي. وعندما تقوم بإدخال أحد الأرقام الصحيحة ما بين ١ و٢٠ في مربع الإدخال يقوم الإجراء بعرض مربع الرسالة الصحيح وفقا للرقم الذي تم إدخاله. لاحظ أن الإجراء لا يقوم بالاختبار لتحديد ما إذا كان القيمة التي تم إدخالها في مربع الإدخال هي بالفعل عدد صحيح بين ١ و٢٠.

ملاحظة يمكن أيضاً تنفيذ التفرع المتعدد عن طريق استخدام عبارات البرمجة On...GoSub وon...GoTo. وقد تم أخذها من الإصدارات السابقة من Basic حيث يستخدم فيها رقم السطر وتسميات السطر لتعيين سطور التعليمات البرمجية. وتتيح بنية Select Case طريقة أكثر سهولة لتنفيذ التفرع المتعدد.

استخدام الحلقات التكرارية للعمليات المتتابعة

في بعض الأحيان تكون هناك حاجة إلى التنقل بين العناصر الموجودة في المصفوفة أو العناصر الموجودة في المجموعة. افترض مثلاً أنك تريد أن تقوم بتغيير النموذج من نموذج إدخال بيانات إلى نموذج مراجعة. ولحفظ عناصر التحكم الموجودة على النموذج فإنك تحتاج إلى الانتقال عبر عناصر التحكم التي تحتوي على البيانات مع إغلاق كل عنصر تحكم يحتوي على البيانات. من إحدى الطرق هي كتابة عبارة منفصلة لكل عنصر تحكم على النموذج الذي يقوم بأداء الاختبار تحديد ما إذا كان عنصر التحكم يحتوي على البيانات ويقوم بإغلاق عنصر تحكم عشوائي. ومن أحد الحلول الجيدة هي استخدام عبارة عنصر التحكم لإخبار VBA لإجراء حلقة عبر عناصر

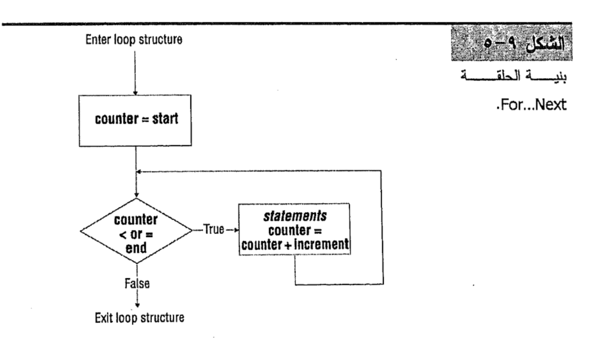
التحكم في النموذج مع تنفيذ العبارة الخاصة بكل عنصر منها. يقدم VBA العديد من الطرق الخاصة ببنيات الحلقات التكرارية التي يمكن استخدامها لتنفيذ نفس مجموعة العبارات البرمجية على نحو متكرر لعدد معين من المرات أو حتى يتم تنفيذ أحد الشروط.

استخدام بنية For...Next

يمكن استخدام بنية For...Next عندما تعلم عدد المرات التي تريد أن تقوم فيها بتنفيذ العبارات البرمجية في الحلقة. يمكن استخدام متغير معاكس لحفظ تتبع التكرارات مع تحديد قيم البداية والنهاية الخاصة بالمتغير المعاكس. يمكن أن تكون وسائط البداية والنهاية والزيادة قيماً رقمية أو تعبيرات. والصيغة الخاصة ببنية الحلقة For...Next هي:

```
For counter = start To end [Step increment]
   statements
Next [counter]
```

Figure 9.5 illustrates the For...Next loop structure.



الشكل ٩-٥
بنية الحلقة For...Next.

ولفهم الطريقة التي تعمل بها الحلقة افترض أن قيمة الزيادة موجبة أو صفرية. يضبط VBA المتغير المعاكس على قيمة البداية عند بدء إحدى الحلقات التكرارية كما أنه يقوم بأحد الإجراءات التالية:

- إذا كان المعاكس أكبر من قيمة النهاية فإن VBA يقوم بالانتقال إلى خارج الحلقة مع تنفيذ العبارة التالية التي تتبع العبارة Next.

- إذا كان المعاكس أقل من أو يساوي قيمة النهاية، فإن VBA ينفذ العبارة الموجودة في الحلقة مع زيادة المتغير المعاكس بنفس قيمة وسيطة الزيادة. وإذا قمت بحذف جملة Step فإن VBA يقوم بزيادة المعاكس بالقيمة واحد.

ينتقل VBA مرة أخرى بعد تنفيذ العبارات إلى عبارة For ثم يقوم بإعادة العملية. وتتكرر الحلقة حتى يصبح المعاكس أكبر من قيمة النهاية. وعندما تكون الزيادة موجبة فإن قيمة النهاية يجب أن تكون أكبر من قيمة البداية وإلا فإن الحلقة سوف تتكرر على نحو لا نهائي.

وفيما يلي مثال لأحد الزيادات الموجبة:

```
For Count = 2 To 10 Step 2
  statements
Next Count
```

يبدأ VBA الحلقة في البنية عن طريق تعيين Count على ٢ ثم يبدأ التمرير الأول إلى الحلقة. ثم يقوم VBA بتنفيذ العبارات مع زيادة Count بالقيمة ٢ والعودة إلى العبارة For لمقارنة قيمة Count بعشرة. وطالما أن قيمة Count أقل من أو تساوي ١٠ فإن VBA يكون له تمرير عبر الحلقة.



ملاحظة: على الرغم من أنه يمكن تعيين مقدار الزيادة على الصفر مع تغيير قيمة المعاكس في عبارات الحلقة، إلا أنه يجب تجنب ذلك لأن ذلك يؤدي إلى صعوبة تصحيح التعليمات البرمجية.

وإذا كانت الزيادة هي رقم سالب فإن VBA يقوم بتعيين المعاكس على قيمة البداية واختبار قيمة المعاكس مع الاختيار بين هذين البديلين:

- إذا كان المعاكس أقل من قيمة النهاية فإن VBA يقوم بالانتقال إلى خارج الحلقة مع تنفيذ العبارة التالية التي تتبع العبارة Next.
- إذا كان المعاكس أكبر من أو يساوي قيمة النهاية، فإن VBA ينفذ العبارة الموجودة في الحلقة مع زيادة المتغير المعاكس بنفس قيمة وسيطة الزيادة.

ينتقل VBA مرة أخرى بعد تنفيذ العبارات إلى عبارة For ثم يقوم بإعادة العملية. وتتكرر الحلقة حتى يصبح المعاكس أقل من قيمة النهاية.ويجب أن تكون قيمة النهاية أقل من قيمة البدء وفيما يلي مثال على ذلك:

```
For Count = 7 to 3 Step -1
  statements
Next Count
```

ملاحظة يمكن حذف المعاكس في عبارة Next، إلا أن التعليمات البرمجية الخاصة بك سوف تصبح أسهل عند القراءة إذا ما تم تضمين المعاكس.

ادرج إجراء Counting الموضح فيما يلي في وحدة basControlStructure لاستكشاف بنية الحلقة For...Next:

```
Public Sub Counting()
    Dim iCount As Integer
    For iCount = 1 To 10 Step 2
        Debug.Print iCount
    Next iCount
End Sub
```

ويعرف الإجراء iCount على أساس أنه المعاكس. وتبدأ الحلقة عن طريق إعداد متغير iCount على القيمة ١ مع طباعة القيمة في الإطار الحالي. ويقوم VBA بزيادة المعاكس بالقيمة ٢ مع تشغيل الحلقة مرة أخرى. ويستمر في ذلك خمسة مرات، وعندما يحتوي المعاكس على القيمة ١١ فإن VBA يخرج من الحلقة.

اكتب Counting في الإطار الحالي مع الضغط على Enter لاختبار الإجراء. يقوم VBA بطباعة الأرقام الصحيحة الفردية من ١ إلى ٩ في الإطار الحالي.

يمكن استخدام البنية For...Next لتنفيذ الحلقة عبر العناصر الخاصة بالمجموعة. وكل مجموعة هي عبارة عن مصفوفة كما أن لها خاصية Count تعيد العناصر الموجودة في المصفوفة. ويمكن الإشارة إلى أحد العناصر الموجودة في المصفوفة عن طريق الموضع الخاص به باستخدام رقم الفهرس. ولأن مجموعات الكائن تكون مبنية على الصفر في أكسس، فإن الفهرس يبدأ بالرقم صفر كما أن يستخدم Count-1 كنهاية له. فمثلاً تكون Forms.Count هي عدد النماذج المفتوحة في الوقت الحالي في قاعدة البيانات. وتكون Forms(0) هي أول النماذج المفتوحة.

ومن أحد الأمثلة على استخدام For...Next مع المجموعات ادرج الإجراء NameForm الموضح فيما يلي في وحدة basControlStructure:

```
Public Sub NameForms()
    Dim iCount As Integer
    For iCount = 0 To Forms.Count - 1
        Debug.Print Forms(iCount).Name
    Next iCount
End Sub
```

يستخدم الإجراء NameForm بنية الحلقة For...Next مع رقم الفهرس الخاص بالمجموعة كمعاكس الحلقة لذكر الأسماء الخاصة بالنماذج المفتوحة في الوقت الحالي. افتح بعض النماذج في وضع Form أو Design ثم اكتب NameForm في الإطار الحالي مع ضغط Enter. فيقوم VBA بذكر كل النماذج المفتوحة.

استخدام بنية For Each...Next

تتشابه حلقة For Each...Next مع الحلقة For...Next إلا أنه بدلاً من إعادة العبارات بعدد معين من المرات تقوم الحلقة For Each...Next بإعادتها مرة واحدة لكل عنصر في المصفوفة أو مع كل كائن في المجموعة. وتكون في غاية القوة والفائدة لأنها تتيح تنفيذ الحلقات التكرارية عبر المجموعات أو المصفوفات بدون الحاجة إلى معرفة عدد العناصر الموجودة. والصيغة هي:

```
For Each element In group
   statements
Next [element]
```

ويكون Group هو اسم مجموعة الكائنات الموجودة في المصفوفة. ويكون element هو المتغير الذي يمثل الكائن في المجموعة كما أنه يجب أن يكون له نوع بيانات Variant أو anObject.

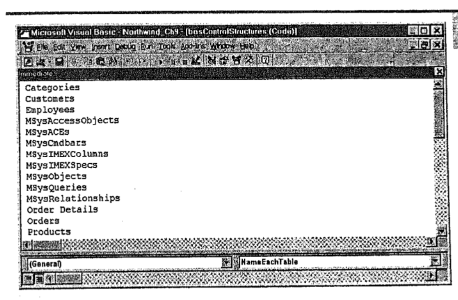
يمكن استخدام بنية For Each...Next للعمل مع المجموعات الخاصة بكائنات Application وكائنات البيانات. وعند العمل مع مجموعات كائنات البيانات يجب استخدام المرجع التام لها. وإذا كان هذا المرجع هو قاعدة البيانات الحالية فإنه يمكنك استخدام وظيفة CurrentDB للإشارة إلى قاعدة البيانات الحالية فمثلاً تشير CurrentDB.TableDefs إلى مجموعة تعريفات الجدول في قاعدة البيانات الحالية. يمكن استخدام بنية الحلقة For Each...Next للتعامل مع كل عضو في المجموعة.

واستخدام بنية For Each...Next لعمل الحلقة بين المجموعة أسرع من استخدام بنية For...Next. والسبب في ذلك أنه عند استخدام بنية For...Next فإن VBA يقوم بالبحث عن العنصر باستخدام رقم الفهرس. ويجب أن يختبر القائمة بأكملها حتى يتم العثور على العنصر المطلوب. أما عند استخدام For Each...Next فإن VBA يتذكر موضعه في القائمة كما أنه ثم يتحرك للأمام بمقدار عنصر واحد مع كل تكرار للحلقة. والوقت الوحيد الذي سوف تحتاج فيه إلى استخدام بنية For...Next هو عند استخدام الحلقة لإزالة العناصر من المجموعة. وفي هذه الحالة قد يفقد VBA موضعه مع النتائج غير المتوقعة إذا استخدمت بنية For...Next.

توضح الأمثلة التالية كيفية استخدام بنية For Each...Next للحصول على القيم الخاصة بالخصائص أو تعيينها وتنفيذ الطرق. المثال الأول هو الإجراء NameEachTable الموضح فيما يلي والذي يمكن إدراجه في وحدة basControlStructure:

```
Public Sub NameEachTable()
    Dim tbl As TableDef
    For Each tbl in CurrentDB.TableDefs
        Debug.Print tbl.Name
    Next tbl
End Sub
```

يستخدم هذا الأجراء بنية For Each...Next لطبع اسم كل جدول في قاعدة البيانات. وبعد كتابتها اكتب NameEachTable في الإطار الحالي مع الضغط على Enter. ويقوم VBA بطباعة اسم كل جدول في قاعدة البيانات بما في ذلك جداول النظام المخفية (انظر الشكل ٩-٦).



الشكل ٩-٦

يستخدم الإجراء NameEachTable بنية الحلقة For Each...Next لطباعة الأسماء الخاصة بالجداول في قاعدة بيانات Northwind.

ثم ادخل إجراء وظيفة FontToRed الموضحة فيما يلي في وحدة basControlStructure.

```
Public Function FontToRed()
    Const Red As Integer = 255
    Dim ctl As Control
    For Each ctl in Screen.ActiveForm.Controls
        ctl.ForeColor = Red
    Next ctl
End Function
```

يغير الإجراء FontToRed لون الخط لكل عناصر التحكم على النموذج إلى اللون الأحمر. وقد تم تصميم هذا الإجراء كإجراء وظيفة عام الذي يمكن أن يعمل مع أي نموذج والذي تحتوي فيه كل عناصر التحكم على خاصية ForeColor الخاصة بكائن Screen للإشارة إلى النموذج والإشارة إلى مجموعة Controls على هيئة Screen.ActiveForm.Controls ، (يمكن أيضاً الإشارة إلى المجموعة بالشكل Screen.ActiveForm بدلاً من Screen.ActiveForm.Controls وذلك لأن Controls هي المجموعة الافتراضية لكائن Form).

افتح نموذج Customers في وضع Design بعد إدخال الإجراء. قم بإضافة زر الأمر المسمى cmdRed بالاسم Change Font Color to Red. انقر زر خاصية OnClick مع تعيين الخاصية على FontToRed()=. احفظ النموذج ثم انتقل إلى وضع Form ثم انقر إلى الزر. يتغير لون الخط في كل عناصر التحكم إلى اللون الأحمر.

استخدام بنية Do...Loop

تتيح لك بنية Do...Loop إعادة العبارات بعدد لا حصر له من المرات. وتستخدم أحد الشروط لتحديد نهاية. ويجب أن يكون تعبيراً يقوم بالتقييم باستخدام True أو False. ويوجد أربع إصدارات من بنية Do...Loop، ويعتمد الإصدار الذي تستخدمه على ما إذا كنت تريد تقييم الشرط في بداية أو نهاية الحلقة وإذا ما كنت تريد أن تستمر الحلقة إذا ما كان الشرط True أو False.

- تستمر Do While...Loop طالما أن الشرط True.
- تنفذ Do...Loop While عبارات الحلقة مرة واحدة ثم تستمر إذا كان الشرط True.
- تستمر Do Until...Loop طالما أن الشرط False.
- تنفذ Do...Loop Until عبارات الحلقة مرة واحدة ثم تستمر إذا كان الشرط False.

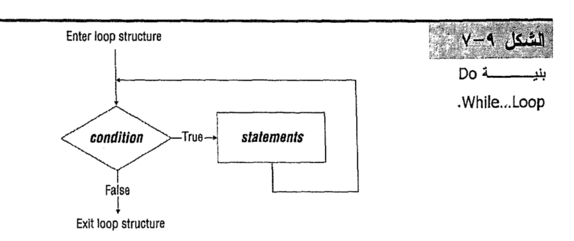
بنية Do While...Loop

وتكون صيغة بنية Do While...Loop

```
Do While condition
   statements
Loop
```

ويبدأ VBA تنفيذ الحلقة عن طريق اختبار الشرط. وإذا كانت قيمة الشرط هي False فإنه ينتقل في الحال إلى خارج الحلقة مع التغاضي عن مع تنفيذ العبارة التي تتبع عبارة Loop. ولا

يتم تنفيذ عبارة Loop إذا كان الشرط False. وإذا كانت قيمة الشرط هي True فإن VBA يقوم بتنفيذ العبارات مع العودة إلى عبارة Do While واختبار الشرط. ويوضح الشكل ٩-٧ بنية Do While...Loop.



الشكل ٩-٧
بنية Do While...Loop.

يستخدم الإجراء DoWhile الموضح فيما يلي بنية Do While...Loop. أضف هذا الإجراء إلى وحدة basControlStructure.

```
Public Sub DoWhile()
   Dim Counter As Integer
   Counter = 0
   Do While Counter < 3
      Debug.Print Counter
      Counter = Counter + 1
   Loop
End Sub
```

قم بتشغيل الإجراء DoWhile في الإطار الحالي. ويتم تشغيل الحلقة ثلاث مرات.

بنية Do While...Loop

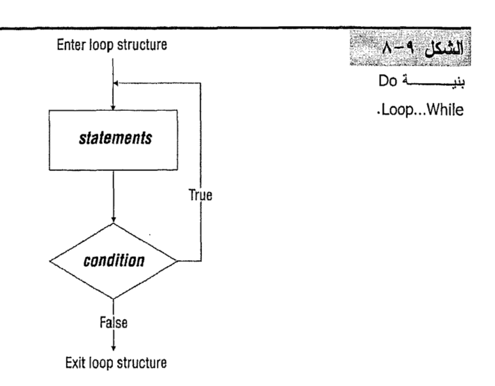
عند وضع جملة الشرط While في نهاية البنية يقوم VBA بتنفيذ العبارات الموجودة في داخل الحلقة مرة واحدة على الأقل.

```
Do
   statements
Loop While condition
```

يقوم VBA بتنفيذ بنية Do Loop...While عن طريق تنفيذ عبارات الحلقة أولاً وبعد ذلك ينتقل VBA إلى عبارة Loop While مع اختبار الشرط. وإذا كانت قيمة الشرط هي False فإن

VBA يخرج من الحلقة مع تنفيذ العبارة التالية التي تلي عبارة Loop While. وإذا كانت قيمتها هي True فإن VBA يقوم بالانتقال إلى البداية مع تنفيذ عبارات الحلقة مرة ثانية. يوضح الشكل ٩-٨ بنية Do Loop...While.

الشكل ٩-٨ بنية Do .Loop...While



يستخدم الإجراء DoLoopWhile الموضح فيما يلي بنية Do Loop...While. أضف هذا الإجراء إلى وحدة basControlStructure.

```
Public Sub DoLoopWhile()
    Dim Counter As Integer
    Counter = 0
    Do
        Debug.Print Counter
        Counter = Counter + 1
    Loop While Counter < 3
End Sub
```

قم بتشغيل الإجراء DoLoopWhile في الإطار الحالي وتستمر الحلقة ثلاثة مرات.

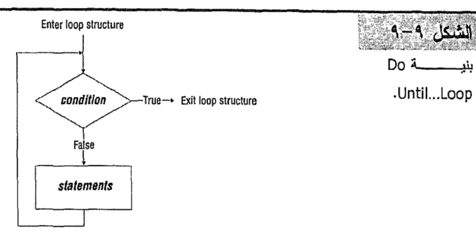
بنيات Do Until...Loop وDo...Loop Until

عند استبدال الكلمة الأساسية While بالكلمة Until، فإن VBA يستمر في تنفيذ الحلقة حتى تصبح قيمة الشرط True.

وتكون الصيغة الخاصة بها على هذا النحو:

```
Do Until condition
   statements
Loop
```

إذا كان الشرط False فإن VBA يقوم بتنفيذ عبارات الحلقة مع العودة إلى عبارة Do. وإذا كان الشرط True فإن VBA يخرج من الحلقة مع تنفيذ العبارة التي تلي عبارة Loop. وإذا كان الشرط True من البداية فإن عبارة بنية Do Until...Loop لا يتم تنفيذها مطلقاً. يوضح الشكل ٩-٩ بنية Do Until...Loop.



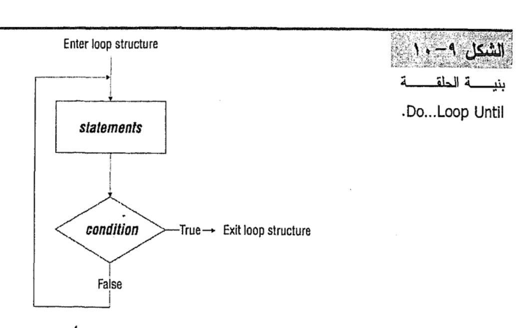
الشكل ٩-٩
بنية Do Until...Loop.

وتكون صيغة بنية Do Until...Loop على هذا النحو:

```
Do
   statements
Loop While condition
```

ينفذ VBA العبارات مع اختبار الشرط. وإذا كان الشرط False فإن VBA ينتقل إلى عبارة Do. وإذا كانت True فإن VBA يخرج من الحلقة مع الاستمرار مع العبارة التالية. إذا كان الشرط هو True من البداية فإن عبارة بنية Do...Loop Until يتم تنفيذها مرة واحدة فقط. يوضح الشكل ٩-١٠ بنية Do...Loop Until.

تحذير يجب التأكد من أن عبارة واحدة على الأقل في Do...Loop تغير القيمة الخاصة بالشرط مما يؤدي إلى جعل الشرط False. وإلا سوف تتكرر الحلقة إلى ما لا نهاية.كما يمكن إيقاف أغلب الحلقات التكرارية اللانهائية عن طريق Ctrl+Break. ويمكن إنشاء حلقة غير منتهية تسمى *tight loop* مما يؤدي إلى إغلاق الحاسب فتكون الطريقة الوحيدة لإنهاء الحلقة هي ضغط Ctrl+Alt+Del والخروج من أكسس.



الشكل ٩-١٠ بنيـــــة الحلقـــــــة .Do...Loop Until

يستخدم الإجراء DoUntil الموضحة فيما يلي بنية Do Until...Loop. أضف هذا الإجـــراء إلى وحدة basControlStructure.

```
Public Sub DoUntil()
    Dim Counter As Integer
    Counter = 0
    Do Until Counter < 3
        Debug.Print Counter
        Counter = Counter + 1
    Loop
End Sub
```

قم بتشغيل الإجراءDoUntil في الإطار الحالي، ولا يتم تنفيذ الحلقة مطلقاً لأن الشـــرط True من البداية.

يستخدم الإجراء DoLoopUntil الموضح فيما يلي بنيـــة Do...Loop Until. أضـــف هـــذا الإجراء إلى وحدة basControlStructure.

```
Public Sub DoLoopUntil()
    Dim Counter As Integer
    Counter = 0
    Do
        Debug.Print Counter
        Counter = Counter + 1
    Loop Until Counter < 3
End Sub
```

قم بتشغيل الإجراء DoLoopUntil في الإطار الحالي، والشرط True لذلك يخرج VBA من الحلقة.

ملاحظة يمكن تنفيذ الحلقة بطريقة أخرى عن طريق بنية While...Wend الذي يمكنك من تحديد شرط الاختبار في بداية الحلقة فقط. إلا أن Do...Loop تتميز بأنها أكثر مرونة لأنه يمكن تحديد شرط الاختبار في البداية أو النهاية.

تنفيذ الحلقة عبر مجموعة السجلات

يمكن استخدام Do Loop للانتقال بين السجلات. ويشير كل تمرير عبر السجلات إلى أحد السجلات. يمكن استخدام خاصية EOF لتحديد مكان انتهاء الحلقة، وتكون False طالما كنت تشير إلى السجل وتكون True إذا قمت بالانتقال بجانب آخر سجل. وعندما تكون في آخر سجل في مجموعة السجلات فإن طريقة MoveNext تقوم بتحريك مؤشر السجل الحالي وراء حدود مجموعة السجلات. كما أن خاصية EOF تصبح True وتنتهي الحلقة.

وبطبيعة الحال فإنك تحتاج إلي تنفيذ عبارات الحلقة إذا كانت مجموعة السجل تحتوي على سجلات بالفعل. لذلك فإنه دائماً ما نستخدم Do Until...Loop للانتقال بين السجلات. وإذا كانت مجموعة السجلات لا تحتوي على أية سجلات فإن خاصية EOF تكون True ولا يتم تنفيذ الحلقة. وبنية الانتقال بين السجلات هي:

```
Do Until rst.EOF
   statements acting on the current record
   rst.MoveNext
Loop
```

ويعتبر rst متغير للكائن يمثل كائن مجموعة السجلات.

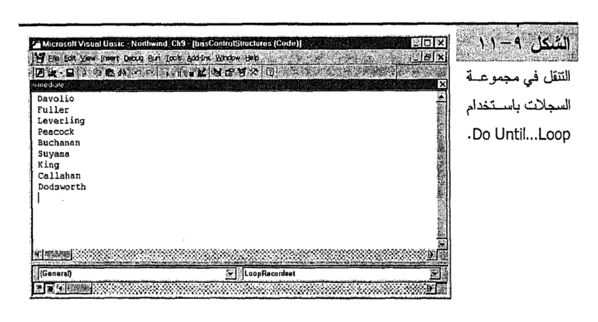
سوف يتم استخدام Do Until...Loop للانتقال بين السجلات في جدول Employee كمثال على ذلك. ادخل الإجراء LoopRecordset الموضح فيما يلي في وحدة basControlStructure:

```
Public Sub LoopRecordset()
   Dim db As Database, rst As Recordset
   Set db = DBEngine(0)(0)
   Set rst = db.OpenRecordset("Employees", dbOpenTable)
```

```
    Do Until rst.EOF
        Debug.Print rst!LastName
        rst.MoveNext
    Loop
End Sub
```

ويوضح هذا الإجراء متغيرات الكائن للإشارة إلى قاعدة البيانات وفتح مجموعة السجل واستخدام المرجع DBEngine(0)(0) للإشارة إلى قاعدة البيانات الحالية. كما يستخدم طريقة OpenRecordset الخاصة بكائن قاعدة البيانات لفتح مجموعة السجل من نوع الجدول في جدول Employee.

قم بتشغيل الإجراء LoopRecordset في نافذة الإطار الحالي. ويؤدي كل تمرير عبر حلقة Do Until إلى طباعة القيمة الموجودة في حقل LastName من السجل الحالي كما يحرك المؤشر الحالي إلى السجل التالي انظر الشكل ٩-١١. وعندما يتم تحريك مؤشر السجل الحالي إلى آخر سجل في مجموعة السجلات تكون خاصية EOF قيمتها True وتنتهي الحلقة.



الشكل ٩-١١ التنقل في مجموعة السجلات باستخدام Do Until...Loop.

تضمين بنيات عناصر التحكم

يمكن وضع بنيات التحكم في بنيات التحكم الأخرى. وفي هذه الحالة يتم تضمينه فيه. افترض على سبيل المثال أنك تريد إغلاق كل مربعات النص الموجودة في النموذج. ويمكن استخدام بنية For Each...Next للانتقال عبر كل عناصر التحكم الموجودة على النموذج. استخدم بنية TypeOf...Then...Else لتحديد ما إذا كان عنصر التحكم هو مربع نصي مع إغلاقه إذا كان هكذا لكل عنصر تحكم.

ويتضمن الإجراء LockControl الموضح فيما يلي بنيات عناصر التحكم لإغلاق المربعات النصية. ويتم تضمين بنية القرار If...Then في بنية الحلقة Each...Next. ويستخدم الإجراء Screen.ActiveForm.Controls للإشارة إلى مجموعة عنصر التحكم الخاصة بالنموذج النشط. ادخل الإجراء في وحدة basControlStructure.

```
Public Function LockControls()
    Dim ctl As Control
    For Each ctl In Screen.ActiveForm.Controls
        If TypeOf ctl Is TextBox Then ctl.Locked = True
    Next ctl
End Function
```

افتح نموذج Customers في وضع Design. ومن النموذج ضع زر الأمر المسمى cmdLockControls مع العنوان Lock the Text Boxes. اضبط خاصية OnClick الخاصة بالزر على ()LockControls=. احفظ النموذج مع الانتقال إلى وضع Form ثم قم بتشغيل الإجراء LockControls عن طريق نقر الزر. يقوم VBA بإغلاق كل عناصر تحكم مربعات الحوار.

اختصار مراجع الكائن

أحياناً ما تحتاج إلى التعامل مع الكائن بالإضافة إلى ذلك فإنك قد تحتاج إلى توفير بعض عناصر المساعدة المرئية لتوضيح أن النموذج في وضع المراجعة بدلاً من عرض الإدخال. ويمكن عمل ذلك عن طريق تغيير لون الخلفية لمربعات الحوار. ومن أحد الحلول لإعداد مجموعة من الخصائص لأحد الكائنات هو استخدام هذه العبارة لكل خاصية: object.property = value. ويجب أن يبحث VBA عن الكائن ومن الممكن أيضاً استخدام بنية With...End With.

وتتيح بنية With...End With لك تنفيذ مجموعة من العبارات على نفس الكائن بدون الحاجة إلى البحث في الكائن عن كل عبارة، وبدلاً من ذلك يتم استخدام المرجع المختصر كما يلي:

```
With object
    [statements]
End With
```

وقد تحتوي كل عبارة على واحد من الأنواع البسيطة التي تقوم بإعداد الخاصية أو تشغيل الطريقة:

```
.property = value
.method
```

كما يمكن أن تحتوي العبارة على أحد التعبيرات التي تشير إلى الخاصية أو الطريقة.

وعندما يكون *object* هو أحد متغيرات الكائن فإن استخدام بنية With قد لا يؤدي إلى حفظ وقت التنفيذ إلا أنه سوف يحفظ وقت البرمجة لأنك سوف تحتاج إلى إدخال متغير الكائن مرة واحدة فقط. وعندما يكون الكائن هو تعبير يؤدي إلى استعادة الكائن فإن استخدامها يؤدي إلى حفظ الوقت لأن VBA يقيم التعبير مرة واحدة فقط. يقيم VBA التعبير مرة واحدة ثم يقوم بتعيين مرجع إلى الكائن من الداخل. ثم يقوم VBA بوضع متغير هذا الكائن المخفي في مقابلة النقاط في العبارات في بنية With.

يستخدم إجراء الوظيفة ReviewMode الموضح فيما يلي بنية With...End With لإعداد الخصائص Locked وEnabled وForeColor الخاصة بعنصر التحكم.

```
Public Function ReviewMode()
    Const Red As Integer = 255
    Dim ctl As Control
    For Each ctl in Screen.ActiveForm
        If TypeOf ctl Is TextBox Then
            With ctl
                .Locked = True
                .Enabled = False
                .ForeColor = Red
            End With
        End If
    Next
End Function
```

يستخدم الإجراء ثلاث مستويات من عناصر التحكم المضمنة لتغيير أي نموذج من وضع التحرير إلى وضع المراجعة:

- يستخدم إجراء الوظيفة بنية Fore Each...Next لتنفيذ الحلقة عبر عناصر التحكم في النموذج النشط.
- يقوم الإجراء بتضمين بنية If TypeOf لتحديد ما إذا كان عنصر التحكم هو مربع نصي من عدمه.
- إذا كان عنصر التحكم مربع نصي فإن الإجراء يستخدم بنية With...End With لتغيير خاصية عنصر التحكم.

وينتقل الإجراء في عناصر التحكم على النموذج النشط، وهي المجموعة الافتراضية لذلك فإن الإجراء يستخدم Screen.ActiveForm للإشارة إلى المجموعة. ويحدد الإجراء إذا ما كان

عنصر التحكم هو مربع نصي وذلك لكل عنصر تحكم وإذا كان كذلك فإنه يغير خصائص عنصر التحكم وإلا فإنه ينتقل إلى عنصر التحكم الآخر.

ادخل الإجراء ReviewMode في وحدة basControlStructure. افتح Customers من وضع Design ثم أضف زر الأمر المسمى cmdReviewMode الذي له التسمية Change to Review Mode. عين زر خاصية OnClick على ReviewMode()=. احفظ النموذج ثم انتقل إلى وضع Form ثم انقر الزر لتشغيل الإجراء. ويقوم الإجراء بإغلاق مربعات النص مع تعيين لون الخط إلى اللون الأحمر.

بعض العبارات والوظائف المفيدة

يقدم VBA مجموع متميزة من العبارات والوظائف المضمنة التي تكون في غاية الفائدة عند الشروع في كتابة الإجراءات وهي كما يلي:

استخدام عبارات التبديل

تحتوي العديد من خصائص الكائن على القيمة True أو False. وتقوم عبارة التبديل بتغيير القيمة الحالية من إحداهما إلى القيمة الأخرى. ويمكن استخدام بنية If...Then...Else كعبارة تبديل كما يلي:

```
If object.property Then object.property = False Else object.
  property = True
```

ويقوم VBAباختبار خاصية *object.property* وإذا كانت التعبير هي True فإن VBA يغيرها إلى False وإلى فإنه يغيرها إلى True.

وتستخدم عبارة التبديل التالية والتي تتميز بأنها أقصر عامل Not:

```
object.property = Not object.property
```

وهو يعكس العبارة لذلك فإنها تعين object.property إلى القيمة العكسية لها فمثلاً العبارة ctl.Locked = Not ctl.Locked تؤدي إلى تبديل خاصية Locked الخاصة بعنصر التحكم.

ويقوم الإجراء ToggleMode الموضح فيما يلي إلى تبديل أي نموذج بين وضع المراجعة ووضع التحرير. وهذا الإجراء يقوم بتبديل عنصر تحكم مربع النص، إلا أنه يمكن تعديل الإجراء ليقوم اختبار والتبديل في أنواع أخرى من عناصر التحكم. ادرج الإجراء ToggleMode في وحدة basControlStructure.

```
Public Function ToggleMode()
    Dim ctl As Control
    For Each ctl in Screen.ActiveForm
        If TypeOf ctl Is TextBox Then
            With ctl
                .Locked = Not .Locked
                .Enabled = Not .Enabled
            End With
        End If
    Next ctl
End Function
```

افتح نموذج Customers في وضع Design. ثم أضف زر الأمر المسمى cmdToggleMode المسمى Toggle the Mode. عين خاصية OnClick على =ToggleMode(). احفظ النموذج ثم انتقل إلى وضع Form ثم انقر الزر عدة مرات لتشغيل الإجراء ToggleMode. ويقوم الوضع بالتبديل بين وضع المراجعة ووضع إدخال البيانات.

استخدام عبارات Exit

يقدم VBA مجموعة من عبارات Exit التي تمكنك من الخروج من الإجراء الخاص بإحدى الوظائف أو الوظائف الفرعية أو العبارات المتكرر مثل Do...Loop أو For...Next أو Each...Next.

وهي تعمل كما يلي:

- تقوم كل من Exit Function وExit Sub وExit Property بالخروج من الوظيفة أو الوظيفة الفرعية أو إجراء الخاصية.
- يمكن أن يستخدم Exit Do مع الحلقات التكرارية For...Next أو Each...Next كما أنه يقوم بنقل عنصر التحكم إلى عبارة Next التالية.
- يمكن أن تستخدم Exit For مع For...Next أو مع Each...Next كما أنها تقوم بتحويل عنصر التحكم في الحال إلى العبارة التي تلي عبارة Next.

استخدام وظيفة Timer

يمكن استخدام وظيفة Timer لتحديد وقت إحدى العمليات في الإجراء. حيث تقوم باسترجاع عدد الثواني بداية من منتصف الليل كما أنها توفر إمكانية حساب المدة التي تستغرقها إحدى العمليات بالتقريب. قم بوضعها في الحال قبل وبعد العملية التي تريد حساب الوقت لها.

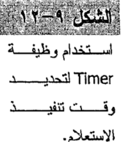
يوضح الإجراء QueryRunTime الموضح فيما يلي عدد الثواني المطلوبة لتشغيل أحد الاستعلامات.

```
Public Function QueryRunTime(strQueryName As String) As Single
    Dim sngBegin As Single, sngEnd As Single
    sngBegin = Timer
    DoCmd.OpenQuery strQueryName
    sngEnd = Timer
    QueryRunTime = sngEnd - sngBegin
    MsgBox strQueryName & " run time is " & QueryRunTime
End Function
```

ويحتوي هذا الإجراء على اسم الاستعلام كوسيطة متسلسلة، كما أنه يوضح المتغير من نوع بيانات Single.

ادرج إجراء الوظيفة QueryRunTime في وحدة basControlStructure. ومن الإطار الحالي اكتب QueryRunTime("Invoices") ثم اضغط Enter. فيقوم الإجراء بتشغيل الاستعلام مع عرض ورقة البيانات الخاصة به وعرض مربع الرسالة.





الشكل ٩-١٢

استخدام وظيفة Timer لتحديد وقت تنفيذ الاستعلام.

استخدام وظيفة DoEvents

هناك نوعان من العمليات في أكسس:

- عملية يتم فيها إرسال واستقبال رسائل ويندوز من وإلى كائنات أكسس.
- عملية لا يتم فيها إرسال واستقبال رسائل ويندوز.

فمثلاً يقوم أكسس بإرسال مدخلات لوحة المفاتيح ونقرات الماوس إلى ويندوز على شكل رسائل. إلا أن VBA لا يقوم بعمل ذلك. وتصطف الرسائل التي لم يتم إرسالها إلى في صف كما أن ضغطات المفاتيح تصطف أيضا في صف SendKeys.

ومن أحد الحلول لمشكلة الرسائل التي لم يتم إرسالها هو استخدام وظيفة DoEvents لتمريـر التحكم إلى الويندوز فيتم التعامل مع الرسائل والمفاتيح، وعند استخدامها فـإن التحكـم لا يعـود لأكسس حتى ينتهي ويندوز من التعامل مع تلك الرسائل. ويتم تضمينها في حلقة تحتاج إلى وقـت طويل مع استخدام الوظيفة لتجميع حصيلة عنصر التحكم من وقت لآخر.

يجعل إجراء DoEventsLoop التالي جهاز الحاسب الآلي يبدأ في حساب الجذر الـتربيعي لفهرس الحلقة للقيم من ١ إلى مليون.

```
Public Sub DoEventsLoop()
    Dim sngBegin As Single, sngEnd As Single, sngElapsed As Single
    Dim Counter As Long, Root As Double
    sngBegin = Timer
    For Counter = 1 To 1000000
        Root = Sqr(Counter)
    Next
    sngEnd = Timer
    sngElapsed = sngEnd - sngBegin
    MsgBox "Run time is " & sngElapsed
End Sub
```

ادرج الإجراء DoEventsLoop في وحدة basControlStructure كما أنه يقوم بتشـغيل البرنامج في الإطار الحالي. ثم قم بتعديل الإجراء DoEventsLoop حتى ينقل عنصر التحكـم إلى الويندوز مرة كل ١٠٠٠٠٠ حلقة كما يلي:

```
Public Sub DoEventsLoop()
  Dim sngBegin As Single, sngEnd As Single, sngElapsed As Single
  Dim Counter As Long, Root As Double
  sngBegin = Timer
  For Counter = 1 To 1000000
        Root = Sqr(Counter)
        If Counter Mod 100000 = 0 Then DoEvents
  Next
  sngEnd = Timer
  sngElapsed = sngEnd - sngBegin
  MsgBox "Run time is " & sngElapsed
End Sub
```

ومن الحلقة التي تم تعديلها يتم تشغيل وظيفة DoEvents إذا كان المعاكس هو أحد مضاعفات ١٠٠٠٠٠. ويستخدم هذا الإصدار معامل Mod لتحديد متى يتم تمريره. والصيغة الخاصة بمعامل Mod هي:

result = number1 Mod number2

ويسمى الباقي Result بعد تقسيم الرقم الأول على الرقم الثاني (مثل 3 Mod 7=1).

قم بتشغيل الإجراء المعدل.

استخدام وظائف MsgBox وInputBox

وتستخدم للسماح للمستخدم بالتعامل مباشرة مع التطبيق. وتستخدم أغلب الأمثلة الموجودة في هذا الكتاب وظيفة MsgBox لعرض رسالة إلى المستخدم. حيث يختار من مجموعة من الأزرار في مربع الحوار. وعند استخدام وظيفة InputBox يقوم المستخدم بإدخال نص في مربع الحوار.

يمكن إنشاء مربعات حوار مخصصة للتعامل مع هاتين الوظيفتين وإلا فإن كل منهما توفر طريقة سهلة ومبسطة لعرض مربعات الحوار الافتراضية.

وتكون صيغة InputBox كما يلي:

InputBox (prompt, title, default, xpos, ypos, helpfile, context)

- *prompt* وهو ضرورية لتعبير السلسلة التي تم عرضها على شكل رسالة.
- *title* وهو سلسلة تعبير اختياري يتم عرضه في كعنوان للحوار.
- *default* وهو سلسلة تعبير اختيارية يتم عرضها في مربع النص كالاستجابة الافتراضية.
- xpos وypos وهي تعبيرات رقمية اختيارية تحدد المسافة بين الركن الأيسر العلوي من الحوار والركن الأيسر العلوي من الشاشة.
- helpfile وهو تعبير السلسلة الافتراضي الذي يقوم بتحديد ملف Help الذي تريد استخدامه.
- *context* وهو التعبير الرقمي الاختياري الذي يعد رقم سياق Help الخاص بملف Help.
- ويحتوي الحوار الذي تم إنشاؤه باستخدام وظيفة InputBox لتجميع الاستجابات وأزرار Ok وCancel. وعندما يقوم المستخدم بالنقر على أحد الأزرار أو الضغط على Enter فإن القيمة الموجودة في المربع يتم استرجاعها إلى أكسس حيث تخزن في أحد المتغيرات.

يمكن استكشاف وظيفة InputBox في الإطار الحالي. اكتب Type var = InputBox ("Enter your name now.") في ثم اضغط Enter. يعرض VBA مربع حوار الإدخال مع انتظار المستخدم حتى يتعامل معه انظر الشكل ٩-١٣. ادخل اسمك ثم انقر OK، فيقوم VBA بتخزين الإدخال في متغير var اكتب ?var ثم اضغط Enter لطباعة قيمة المتغير.

الشكل ٩-١٣

استخدام وظيفة InputBox لتجميع القيمة.

Enter your name now.

OK

Cancel

استخدام وظيفة SysCmd

تعتبر في واقع الأمر ثلاثة وظائف في وظيفة واحدة. ويمكن استخدامها كما يلي:

- لاستعادة حالة كائن الإطار Database.
- لاسترجاع معلومات نظام أكسس.
- لعرض مقياس التقدم.

استعادة حالة كائن الإطار Database

وهو من أحد أشهر استخدامات هذه الوظيفة حيث يتم تحديد ما إذا كان الكائن مفتوحاً أو جديداً. وصيغته هي:

```
returnvalue = SysCmd(acSysCmdGetObjectState, objecttype,
objectname)
```

ويعد *objecttype* من أحد الثوابت لتحديد أحد كائنات إطار Database acTable وacQuery وacForm وacReport وacMacro وacModule
وacDataAccessPage, وacDefault وacDiagram وacServerView
acStoreProcedure. Objectname كما أنه عبارة عن تعبير السلسلة التي هي الاسم الصالح الذي قد تم تعيينه للكائن. تقوم الوظيفة باسترجاع العدد الصحيح الذي هو عبارة عن خليط من القيم التالية.

القيمة	الثابت الحقيقي	حالة الكائن
٠		غير مفتوح أو لا يوجد
١	AcObjStateOpen	مفتوح Open
٢	AcObjStateDirty	تم تغييره ولكنه لم يحفظ
٤	AcObjStateNew	جديد New

يمكن استكشاف هذا الإصدار من وظيفة SysCmd في الإطار الحالي. ومع إغلاق نموذج Customers اكتب ? SysCmd(acSysCmdGetObjectState, acForm ثم اضغط على Enter. فيقوم الإطار الحالي بكتابة صفر للإشارة إلى أن النموذج ليس مفتوحاً. ثم افتح نموذج Customers في وضع Design مع تغيير خاصية RecordSelector إلى Yes، اكتب ? SysCmd(acSysCmdGetObjectState, acForm, "Customers") ثم اضغط على Enter. فيقوم الإطار الحالي بكتابة ٣ للإشارة إلى أن النموذج مفتوح وقدتم تغييره إلا أن هذه التغييرات لم يتم حفظها.

إعادة معلومات نظام أكسس

يمكن استخدام وظيفة SysCmd لاستعادة المعلومات عن أكسس مثل رقم الإصدار وموضع الملفات وما إلى ذلك. وتكون الصيغة في هذه الحالة هي:

```
returnvalue = SysCmd(action)
```

وتكون القيمة المسترجعة هي المعلومات وتكون على هيئة سلسلة.

يمكن استكشاف هذه النسخة من وظيفة SysCmd في الإطار التالي على هذا النحو:

- اكتب ?SysCmd(acSysCmdAccessVer) مع الضغط على Enter. فيقوم الإطار الحالي بكتابة رقم الإصدار الحالي من أكسس.
- اكتب ?SysCmd(acSysCmdGetWorkgroupFile) مع الضغط على Enter. فيقوم الإطار الحالي بكتابة مسار ملف مجموعة العمل.
- اكتب ?SysCmd(acSysCmdRuntime) مع الضغط على Enter. فيقوم الإطار الحالي بكتابة True أو False حسب ما إذا كان الإصدار خاص بوقت التشغيل (يتيح لك Developer's Kit إنشاء إصدارات وقت التشغيل فقط من أكسس).
- اكتب ?SysCmd(acSysCmdAccessDir) مع الضغط على Enter. فيقوم الإطار الحالي بكتابة اسم الفهرس الذي يحتوي على Msacces.exe.

- اكتب ?SysCmd(acSysCmdProfile) مع الضغط على Enter. فيقوم الإطار الحالي بكتابة الإعداد الجانبي/المحدد عند بدء تشغيل أكسس من سطر الأمر.

عرض مقياس التقدم

يمكن استخدام وظيفة SysCmd لعرض مع النص أو عرض رسالة النص في شريط الحالة للإشارة إلى التقدم الخاص بالعملية وتكون الصيغة:

```
returnvalue = SysCmd(action[,text][,value])
```

حيث:

- action هو الثابت الحقيقي الذي يمكنك استخدامه لتحديد الإجراء المطلوب.
- text هو الوسيطة الاختيارية لتحديد النص المراد عرضه.
- value وهو الوسيطة الاختيارية لتحديد القيمة العظمى لمقياس التقدم عند استخدام الوظيفة لبدء العد ولتحديد القيمة النسبية للمقياس عند استخدام الوظيفة لتحديث المقياس.

لعرض مقياس التقدم يجب أن تستخدم الوظيفة لتحديد المقياس الأولي لتحديد الإجراء مع الثابت الحقيقي acSysCmdInitMeter. ولعرض تقدم إحدى العمليات استخدم الوظيفة لتحديد القيمة الحقيقية acSysCmdUpdateMeter والقيمة كمقياس للتقدم. تقوم الوظيفة SysCmd بحساب نسبة التقدم للقيمة العليا مع تحديث المقياس.

لاستكشاف هذا الإصدار من وظيفة SysCmd ادخل الإجراء Meter الموضح فيما يلي في وحدة basControlStructure.

```
Public Function Meter()
    Dim varReturn As Variant, str As String, Counter As Long
    Dim Root As Double
    str = "Calculating square roots..."
    varReturn = SysCmd(acSysCmdInitMeter, str, 1000000)
    For Counter = 1 To 1000000
        Root = Sqr(Counter)
        If Counter Mod 100000 = 0 Then
            varReturn = SysCmd(acSysCmdUpdateMeter, Counter)
        End If
    Next Counter
    varReturn = SysCmd(acSysCmdRemoveMeter)
End Function
```

يستخدم هذا الإجراء حلقة For...Next لحساب الجذر التربيعي للأعداد بين ١ و١٠٠٠٠٠٠. وقبل البدء في الحلقة فإن الإجراء يقوم بعرض مقياس التقدم الأولي. مع تعيين القيمة العظمى على ١٠٠٠٠٠٠ ويقوم الإجراء بتحديث مقياس التقدم باستخدام قيمة متغير Counter كل ١٠٠٠٠٠ حلقة. وعند انتهاء الحلقة فإن الإجراء يقوم بإزالة مقياس التقدم.

افتح نموذج Customers في وضع Design مع إضافة زر أمر جديد يسمى cmdSquareRoot مع خاصية Calculate Square Roots Caption . انقر خاصية OnClick ثم اكتب =Meter(). احفظ النموذج ثم انتقل إلى وضع Form. انقر الزر الجديد، فيعرض مقياس مقدار تقدم العملية (انظر الشكل ٩-١٤).

الشكل ٩-١٤

استخدام وظيفة SysCmd لعرض مقياس التقدم الأولي وتحديثه ثم إزالته.

خلاصة

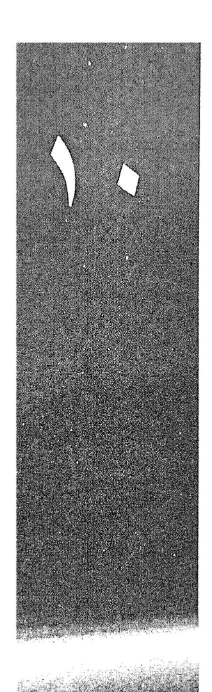
قدم هذا الفصل الكلمات الأساسية والعبارات التي يقدمها VBA للتحكم في ترتيب العبارات الذي يتم فيه تنفيذ تلك العبارات أو ما إذا كانت سوف تنفذ أصلاً من عدمه بالإضافة إلى عدد المرات التي تتكرر فيها.انظر النقاط التالية:

- ♦ استخدم عبارات التحكم الموجودة في بداية ونهاية المجموعة لإنشاء بنية التحكم وللتحكم في تنفيذ مجموعة من العبارات
- ♦ لاتخاذ أحد القرارات قم بإنشاء واحد أو أكثر من شروط الاختبار مع استخدام قرار بنية التحكم لاختبار الشروط وتنفيذ مجموعات مختلفة من العبارات بالاعتماد على قيمة

الشروط. وتكون صيغة بنيات التحكم لقرار التحكم هي If...Then, If...Then...Else, and Select Case.

- بنيات التحكم التي يمكن استخدامها لإعادة مجموعة من العبارات هي For...Next, For Each...Next, and Do...Loop
- يمكن تضمين بنية عنصر تحكم داخل بنية عنصر تحكم آخر لإنشاء نماذج أكثر تعقيداً عند التنفيذ.
- يمكن استخدام بنية With...End With عند إعداد عدة خصائص لنفس الكائن.
- يقدم أكسس العديد من العبارات والوظائف المضمنة التي تجعل البرنامج اكثر فائدة وكفاءة وقوة. ومن الأمثلة على ذلك وظيفة Timer ووظيفة DoEvents التي تقطع تنفيذ الإجراء كما تتيح لويندوز الرد على الرسائل. ووظيفة SysCmd التي تحدد حالة كائن إطار Database مع تجميع المعلومات عن النظام أو عرض مقياس التقدم.

يكمل هذا الفصل مقدمة Access VBA. ألان قد اتضحت المفاهيم والمعايير المطلوبة للتعامل مع قواعد البيانات بشكل آلي، إلا أنه يستحسن دراسة الإمكانات الخاصة بمعالجة الأخطاء في الفصل التالي قبل جعل عمليات معينة في قاعدة البيانات تتم بشكل آلي حيث أنها في غاية الأهمية.



التعامل مع الأخطاء في VBA

تعريف الخطأ بالمعنى العام هو الانحراف عن ما كونه صحيحاً ويحدث أحياناً نتيجة شيء ما لم تكن تنوي فعله. وبناءاً على نوع الخطأ، قد يحدث أي من الآتي:

- قد لا يمكنك تشغيل جزء من التطبيق الخاص بك.
- قد يتم تشغيل التطبيق ولكن عند حدوث الخطأ، يقوم أكسس باستخدام معالج الخطأ الافتراضي بواسطة عرض رسالة خطأ افتراضية وفي بعض الحالات بواسطة إنهاء التطبيق وتعليق تنفيذ إضافي.
- يتم تشغيل التطبيق ولكنه يفشل في تنفيذ العملية التي تقصدها على الرغم أن أكسس لا يعطي أي علامة للمشكلة.

تزود تعليماتVBA بفحص بعض الأخطاء التي يمكن حدوثها. تزود VBA أيضاً بمجموعة أدوات تصحيح الأخطاء قبل حدوثها والتي يمكنك استخدامها في تحليل الخطأ. وبمجرد أن تفهم سبب حدوث الخطأ، يمكنك آنذاك اتخاذ الخطوات اللازمة لتصحيح المشكلة. يمكنك أحياناً تصحيح المشكلة لكي لا يحدث الخطأ ثانية وأحياناً أخرى قد يكون هذا الخطأ ليس من السهل إزالته وعليك آنذاك التعامل مع حقيقة إ'مكانية حدوثه ثانية. وتصحيح هذا النوع من الأخطاء يعني كتابة إجراء VBA للتزويد بالتعليمات حول كيفية معالجة الخطأ عند حدوثه ثانية.

وعند تكوين تطبيقات مفصلة، فمن المؤكد أنك سوف تواجه بعض الأخطاء. والخطوة الأولى في التعامل بعض الأخطاء هي فهم كيفية وسبب حدوث ذلك. يوضح هذا الفصل الأنواع المختلفة للأخطاء وكيف يمكنك استخدام الصيغة المضمنة للفحص والمترجم لتجنب بعضها وكذلك كيفية استخدام أدوات تصحيح الأخطاء قبل وقوعها لتحليل الخطأ وكيف تقوم باستبدال معالج الخطأ الافتراضية برمز معالجة الخطأ الخاص بك.

الأخطاء التي يمكن تجنبها والأخطاء التي لا يمكن تجنبها

سوف تواجه نوعين من الأخطاء وهي التي يمكنك تجنبها والأخطاء التي لا يمكنك تجنبها.

الأخطاء التي يمكن تجنبها

تعتبر الأخطاء التي يمكن تجنبها نتيجة أخطائك. كل واحد يعمل أخطاء وكلما ازدادت خبرتك في البرمجة كلما قلت أخطائك ولكن لن تزال تعمل بعض الأخطاء. وبمعرفة أنواع الأخطاء التي تحد=ث وكيفية توقع حدوثها هي أفضل طريقة لتعلم كيفية تجنبها عند كتابة الرموز. وهناك ثلاثة أنواع من الأخطاء التي يمكن تجنبها:

- أخطاء عند وقت الترجمة وهي التي تحدث عند انحراف قواعد صيغة VBA كما تقوم أحياناً بخطأ في هجاء كلمة أو أن تنسى جملة IF في هيكل قرار If...then. يمكنك

إزالة هذه الأخطاء باستخدام ترجمة وفحص الصيغة المضمنة في VBA. "عيك تمكين فحص الصيغة المضمنة بواسطة فحص خيار Auto Syntax Check في مربع حوار خيارات المحرر للفيجوال بيسك".

- أخطاء عند التشغيل وهي العتي تحدث عندما يكون أكسس غير قادر على تشغيل جملة VBA لأنك قمت بعمل خطأ مثلما تقوم بتحديد نوع بيانات خطأ أو محاولة تشغيل طريقة خاطئة.
- أخطاء اعتبارية وهي التي تحدث عندما تقوم إجراءات VBA بالتنفيذ بدون فشل ولكن بدون الحصول على نتيجة التي قصدت إليها. على سبيل ا لمثال، يحدث الخطأ الاعتباري عندما تقوم بتخصيص إجراء حدث للحدث الخاطئ وتشغيل إجراءين بالترتيب الخاطئ نتيجة لذلك. "تسمى هذه الأخطاء بوجه عام Bugs على الرغم أن Bug تستخدم للإشارة إلى أي خطأ".

يمكنك التعامل مع الأخطاء التي يمكن تجنبها بواسطة إزالة مصدر الخطأ "على سبيل المثال، تصحيح خطأ كتابة"، باستخدام الخصائص والطرق الصالحة للوحدة فقط أو أن تقوم بإطلاق إجراءات الحدث بواسطة الحدث المناسب. يمكنك منع العديد من الأخطاء بواسطة كتابة إجراء VBA بواسطة اختبار قيمة الحالة المسبقة أولاً ثم تنفيذ جملة إذا كانت هذه الحالة المسبقة ذات قيمة True. على سبيل المثال، تفشل قيمة اختيار النموذج إذا كان النموذج غير مفتوح، ولمن يمكنك تجنب الخطأ بواسطة استخدام وظيفة IsLOaded لتحديد ما إذا كان النموذج مفتوح ويقوم بتشغيل الجملة لتحديد النموذج فقط إذا كان النموذج مفتوحاً. وإذا كانت وظيفة IsLOaded ذات قيمة False، يمكنك أن تسبق الجملة بتحديد النموذج بواسطة طريقة OpenForm. ويمكن القيام بذلك بواسطة اختيار قرار If...Then.

الأخطاء التي لا يمكن تجنبها

الأخطاء التي لا يمكن تجنبها هي التي تستمر في الحدوث حتى بعد إزالة كل الأخطاء من خلال تصميم واختبار وتحري أخطاء جيد. وهاك بعض المواقف التي تحدث فيها الأخطاء التي لا يمكن تجنبها:

- محاولة المستخدم حفظ سجل جديد بدون المفتاح الأساسي
- إدخال المستخدم قيمة في المربع القابل للتحرير غير موجودة في القائمة القابلة للتحرير
- فشل آلي أو كون القرص ممتلئ
- قطع الشبكة بدون توقع
- إدخال المستخدم للقرص المطاط بطريقة خاطئ

يمكنك تخيل الأخطاء التي لا يمكن تجنبها كخطاء نتج عن شخص آخر بمعنى أن الكمبيوتر أو المستخدم قام بشيء ما ولا يمكنك إصلاحه. ومع ذلك فما ذال الشأن قائم ولابد أن يقوم الرمز بالتعامل مع الأخطاء التي لا يمكن تجنبها.

الأخطاء الجسيمة والأخطاء البسيطة

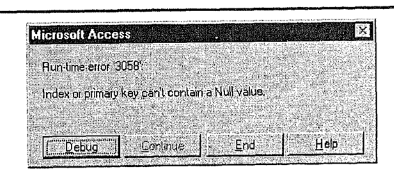
تأتي الأخطاء التي لا يمكن تجنبها على نوعين: تلك التي تسبب فشل الإجراء الذي تقوم به (الأخطاء الجسيمة) وتلك التي لا تسبب فشل (الأخطاء البسيطة).

عند حدوث الأخطاء البسيطة يقوم معالج الخطأ الافتراضي بأكسس بعرض رسالة خطأ افتراضية ويستمر إجراء VBA في حالة التشغيل.على سبيل المثال، في حالة تشغيل طريقة ApplyFilter باسم استعلام غير موجود كمعامل Filtename. فيستمر الإجراء في حالة التشغيل بدون عرض رسالة خطأ المشغل لأن الطريقة المضمنة مصممة بحيث لا يتم تكوين رسالة عند تشغيل VBA.

ولسوء الحظ فإن معظم الأخطاء تكون فادحة. وتحدث الأخطاء الجسيمة عند عدم قدرة جملة مشغل VBA على التنفيذ ولا توجد مقاييس مضمنة تمنع خطأ مشغل VBA من تكوينه. وعند حدوث الخطأ الفادح، تكون معالجة الخطأ الافتراضي هي أن VBA قد يعرض رسالة خطأ افتراضية وقد يقوم أكسس بعرض مربع اختيار خطأ المشغل للإجراء وعليه يقوم أكسس بتعليق التنفيذ. على سبيلاً المثال، افترض أنك تضع زر أمر على نموذج وتقوم بتكوين إجراء حدث يقوم بتشغيل أمر حفظ السجل لحفظ السجل، وهو كالآتي:

```
Private Sub cmdSave_Click()
    DoCmd.RunCommand acCmdSaveRecord
End Sub
```

عندما تقوم بإدخال سجل جديد ولكن تترك المفتاح الأساسي فارغ وتحاول أن تحفظ السجل، فإن محرك قاعدة البيانات لن يكون قادر على حفظ السجل ويقوم أكسس بعرض رسالة خطأ المشغل الموضحة في الشكل ١٠-١ ويقوم كذلك بتعليق تنفيذ الإجراء إلى جملة DoCmd.



الشكل ١٠-١
رسالة خطأ المشغل للخطأ الفادح.

حصر الخطأ

يتم التعامل مع الأخطاء التي لا يمكن تجنبها بواسطة كتابة رمز معالجـــات للخطـــأ التفصيلـــي. وتسمي عملية العثور على الخطأ الافتراضي بحصر الخطأ. يمكنك كتابة معالج الخطأ الخاص بك الذي يقوم باستبدال رسالة الخطأ الافتراضية برسالة الخطأ المفصلة وهذه تتضمن تعليمات إضافية لمعالجة الخطأ. قد تحتاج هذه التعليمات إلى إنهاء الإجراء بدون إنجاز المهمة المقصودة ولكـــن يمكنك على الأقل تضمين التعليمات بإنهاء الإجراء بدون عرض مربع الإجراء الفاشل الذي يقـوم بوضع الإجراء في حالة تنفيذ معلق والمستخدم المبتدئ في حالة خوف من الفهم المعلق.

كتوضيح لمعالجة الخطأ المفصل، يقوم إجراء cmdSave_Click لحفظ الســـجل والموضـــح أدناه بحصر معالج الخطأ الافتراضي والذي سوف يقوم أكسس بتنفيذه عند حدوث الخطأ الفـــادح لأن قيمة المفتاح الأساسي مفقودة. يتضمن الإجراء جمل لحصــــر الخطـــأ واســـتبدال الرســـالة الافتراضية الموضحة في الشكل ١٠-١ بالرسالة المفصلة ونقـــل التركـــيز إلـــى التحكـــم فـــي CustomerID.

```
Private Sub cmdSave_Click()
    On Error GoTo Err_cmdSave_Click
        DoCmd.RunCommand acCmdSaveRecord
Exit_cmdSave_Click:
Exit Sub
Err_cmdSave_Click:
    MsgBox "You must enter a unique Customer ID before saving " _
        & "the record."
    DoCmd.GoToControl "CustomerID"
        Resume Exit_cmdSave_Click
End Sub
```

سوف تتم مناقشة معالجة الخطأ بتفصيل أكثر في نهاية هذا الفصل.

مترجم VBA

VBA لا يقوم باستخدام مترجم حقيقي، فالمترجم الحقيقي يقوم بترجمة الرمز الزي تكتبه في لغـــة برمجة مثل C++,Basic أوPascal، ويسمى هذا الرمز برمز المصدر ويقـــوم بتكويـــن لغـــة منسوخة آلياً للرمز المسمى رمز الوحدة.

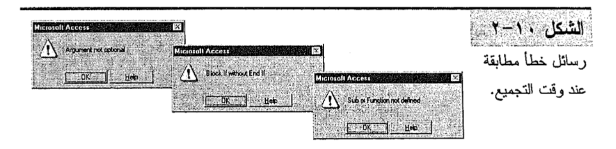
ومميزات استخدام المترجم الحقيقي هي أنه يمكن لرمز الوحدة التشغيل على الكمبيوتر الـــذي به مترجم مركب وأن رمز الوحدة هذا يعمل بشكل أسرع من رمز المصدر غير المترجم.

يقوم مترجم VBA بترجمة الرمز إلى حالة بين رمز المصدر ورمز الوحدة ويسمى هذا الرمز "رمز وهمي". يمكنك عرض أو قراءة الرمز الوهمي ولكن يكون تشغيله أسرع من رمز المصدر "على الرغم أنه ليس أسرع من الرمز المعادل الذي يتم ترجمته بواسطة مترجم حقيقي". فبدلاً من وجود أكسس على جهازك لكي يقوم بتشغيل الرمز الوهمي.

ملاحظة على الرغم من الأخطاء، فغن هذا الكتاب يتبع التطبيق العام وهو الإشارة إلي الرمز الوهمي الذي يتم توليده بواسطة مترجم VBA مثل حالة الترجمة ويشير إلي عملية ترجمة الرمز إلي رمز وهمي كترجمة.

تسمي الأخطاء التي يقوم أكسس بتصحيحها أثناء الترجمة أخطاء وقت الترجمة. تحدث أخطاء ترجمة فعلية تحت الأحوال التالية:

- عند استخدام متغير لم يتم تعريفه أو عند الخطأ في كتابة اسم المتغير (وهذا عندما يكون قسم Declarations متضمن جملة option Explicit)
- عند حذف معامل مطلوب لوظيفة أو طريقة انظر الشكل ١٠-٢ أ
- إذا قمت بحذف الجملة المطلوبة في المجموعة، مثل جملة End If في هيكل التحكم If...Then انظر الشكل ١٠-٢ ب
- إذا قام الإجراء الخاص بك باستدعاء إجراء آخر غير موجود انظر الشكل ١٠-٢ ج



الشكل ١٠-٢ رسائل خطأ مطابقة عند وقت التجميع.

الترجمة الآلية

عندما تقوم بمحاولة تشغيل إجراء متغير أو جديد، يقوم أكسس بترجمة الإجراء آلياً. وافتراضياً، يقوم أكسس بترجمة الرمز الذي يقوم بترجمة الرمز الذي يجب أن يقوم بالترجمة لكي يتم تشغيل الإجراء. يقوم أكسس بترجمة الإجراء الذي تريد تشغيله وكذلك ترجمة أي إجراءات يقوم باستدعائها الإجراء الحالي وهكذا.ومجموعة الإجراءات التي يمكن أن تقوم الإجراءات المخزنة باستدعائها قد تسمى "مسار التنفيذ الحالي" للإجراء الحالي. وبالنسبة لشجرة الاستدعاءات عبارة

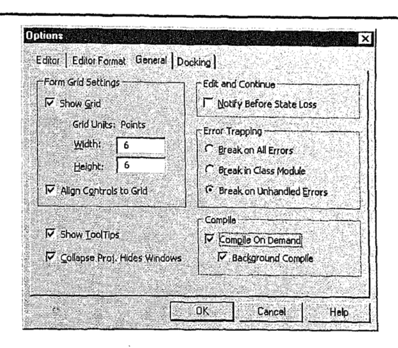
عن مجموعة من كل الوحدات التي تحتوي على إجراءات التي قد يتم استدعائها بواسطة أي إجراء مخزن في وحدة الإجراء النمطية.

إضافة إلى ترجمة الإجراء الحالي والإجراءات التي توجد في شجرة الاستدعاءات الخاصة بها، يقوم أكسس بفحص هذه الإجراءات بحثاً عن أخطاء إشارة إلى المتغيرات التي تؤكد أن كل المتغيرات المشار إليها تم تعريفها بشكل صحيح "وهذا إذا كانت جملة Option Explicit متضمنة في أقسام تعريفات لكل الوحدات النمطية". عند ترجمة الإجراء، يقوم أكسس بفحص المتغيرات التي يستخدمها الإجراء الذي يتم تعريفه في مكان آخر "في قسم Declarations للوحدة النمطية على سبيل المثال".

وأثناء الترجمة، يقوم أكسس بتحميل الوحدات النمطية التي تحوي كل الإجراءات في مسار التنفيذ الحالي وكذلك الوحدات النمطية التي تعرف المتغيرات التي أشرنا إليها في هذه الإجراءات. وفي حالة عدم حدوث أخطاء ترجمة، يقوم أكسس بترجمة الإجراءات في مسار التنفيذ الحالي وتكون نسخة مترجمة من الرمز. وبع الترجمة، عند استدعاء الإجراء، يقوم أكسس بتشغيل النسخة المترجمة. يقوم اكسس أيضاً بتتبع حالة تجميع الإجراء. وبعد الترجمة، لا يقوم أكسس بترجمة الإجراء ثانية إلا إذا قمت بعمل تغيير يسبب إلغاء ترجمة الإجراء.

ملاحظة تسمى النسخة القابلة لتحرير الإجراء التي تم عرضها في نافذة لوحدة النمطية رمز المصدر. وبعد ترجمة الإجراء، يقوم أكسس بتكوين النسخة مترجمة ولا يمكنك عرض أو تحرير النسخة المترجمة. يمكنك حفظ التطبيق الخاص بك في ملف mde الذي يوجد به رمز المصدر الذي تمت إزالته. وفي ملف mde، تكوم جهودك محمية تماماً ولا يمكن لأي شخص مشاهدة أو تغيير الرمز الخاص بك. وعند تكوين ملف mde، تأكد أنك تقوم بالاحتفاظ بنسخة من قاعدة البيانات الأصلية لكي تكون قادر على مشاهدة وتغيير الرمز الخاص بك. لمزيد من المعلومات عن حفظ قاعدة البيانات في mde، انظر الفصل ٧.

يمكنك وضع مستويان من الترجمة الآلية، وهما متاحان عن طريق مربع خيارات محرر فيجوال بيسك. قم بعرض مربح الحوار بواسطة اختيار Tools ⇦ Options ثم انقر على علامة الجدولة General انظر الشكل ١٠-٣. يمكنك هنا تمكين أو تعطيل الترجمة في خيار Demand.



الشكل ١٠-٣
يقوم أكسس بترجمة الإجـــــراءات الموجودة في مسار تنفيذ الإجــراء الحـالي للإجـراء الذي تقوم بتشـغيله في حالــة تمكيـن ترجمة في خيـــار Option.

عند تمكين ترجمة في خيار Demand "الافتراضـي"، يقوم أكســس بالترجمـة الإجـراءات الموجودة في شجرة الاستدعاءات التي تقوم بتشغيلها. وعلى الرغم أن الترجمــة علـى خيـار Demand تجعل كتابة واختبار الإجراءات المفردة أسرعن إلا أنها لا تجد أخطاء قد توجد فـي الوحدات النمطية الخاصة بك. وعند تعطيل هذا الخيار، يقوم أكسس بترجمة كل الإجراءات فـي شجرة الاستدعاءات "سواء كانوا باسم الإجراء الحالي أم لا" وكذلك ترجمة الوحدات النمطية التي تقوم بتعريف المتغيرات المشار إليها في الإجراءات الحالية وشجرة الاستدعاءات.

لاكتشاف Compile على خيار Demand، تأكد أن الخيار تم تمكينه في علامــة الجدولـة Generalفي Options في Visual Basic Editor، ثم اتبع الخطوات التالية:

١- تكوين قاعدة بيانات تسمى CH10_Examples.

٢- تكوين وحدة نمطية تسمى basCompileNumber ثم أدخل الإجراءات التالية:

```
Public Sub TestA()
   MsgBox "This is TestA"
End Sub
Public Sub TestC()
   Call TestD
End Sub
```

٣- يتم تكوين وحدة نمطية ثانية تسمى basCompileAlpha ثم قــم بتكويــن الإجرائييــن التاليين. يكون لإجراء TestC خطأ مجمع لان إجراء TestD لا يوجد. لاحظ أيضــا أن إجراء TestTwo في الوحدة النمطية bas-CompileNumber لها خطأ مجمع ممــاثل

لان إجراء TestB لا يوجد.

```
Public Sub TestOne()
   Call TestA
End Sub
Public Sub TestTwo()
   Call TestB
End Sub
```

٤-في إطار Immediate، اكتب Call TestOne واضغط Enter. يجمع أكسس TestOne وTestA ويعرض مربع رسالة. يسجل أكسس حالة التجميع لإجراءات TestOne وTestA ولن يجمعهم مرة أخرى إلا إذا قمت بتغيير. بسبب اختيار خيار Compile on Demand، لا يقوم VBA بتجميع إجراءات TestTwo وTestC.

٥-اختر Tools ⇦ Options.، حدد تبويب General، وقم بإخلاء خانة اختيار Compile on Demand.

٦-قم بتغيير إجراء TestOne بإضافة تعليق كما موضح فيما يلي لإجبار VBA أن يعيد تجميع إجراء Test1. قم بتشغيل الإجراء في إطار Immediate، ولكن هذه المرة سيحاول VBA تجميع كل الإجراءات في الوحدة النمطية basCompileNumber ويكتشف الخطأ المجمع في إجراء TestTwo. يعرض VBA مربع رسالة الخطأ المجمع (الشكل ١٠-٤أ) ويميز عبارة المشكلة في إجراء TestTwo (الشكل ١٠-٤ب).

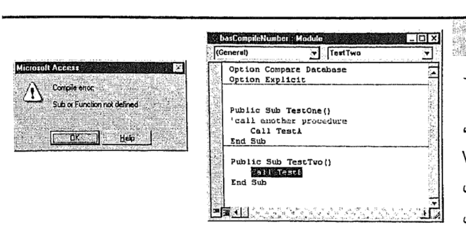
```
Sub TestOne()
   'call another procedure
   Call TestA
End Sub
```

٧-قم بتغيير العبارة التي تم تمييزها إلى Call TestA ثم قم بتشغيل إجراء TestOne في إطار Immediate. هذه المرة تجمع الوحدة النمطية basCompileNumber بنجاح، لكن عندما يحاول أكسس تجميع الوحدة النمطية basCompileAlpha يتم اكتشاف خطأ في إجراء TestC في الوحدة النمطية basCompileAlpha على الرغم من أن هذا الإجراء لا يتم استدعائه من خلال الإجراء الحالي أو من خلال أي من الإجراءات في الوحدة النمطية basCompileNumber.

٨-اختر Tools ⇦ Options،حدد تبويب General، وقم باختيار خيار Compile on Demand.

الشكل ١٠-٤

عند إلغاء اختيار خيار Compile on Demand، سيحاول VBA تجميع كل الإجراءات في وحدة نمطية سواء كانت الإجراءات يتم استدعائها من خلال الإجراء الحالي أم لا.



التجميع الواضح

لتجميع إجراء واحد، يمكن فقط أن تقوم بتشغيله. يمكن أيضا أن تجعل أكسس يقوم بتجميع كل الإجراءات في وحدة نمطية موجودة في قاعدة البيانات.

لتجميع كل الإجراءات في الوحدة النمطية في قاعدة البيانات، اختر Debug ⇦ Compile(database). بعد تشغيل هذا الأمر، تكون قاعدة البيانات في حالة مجمعة مع كل الإجراءات في كل الوحدات النمطية المجمعة وجاهزة للتشغيل. لان التجميع يأخذ، يجب أن تتأكد أن قاعدة البيانات في حالة مجمعة بعد الانتهاء من التغييرات وبذلك تكون جاهزاً لتشغيل قاعدة البيانات الكاملة.

إلغاء التجميع

أي تغييرات للتعليمات البرمجية في الوحدة النمطية يلغي تجميع الوحدة النمطية. التغييرات التي يتم تنفيذها على الكائنات التي لها تعليمات برمجية مخصصة لهم، مثل النماذج، التقارير، وعناصر التحكم وكذلك يتسبب في جعل الوحدة النمطية غير مجمعة. يخزن أكسس اسم قاعدة البيانات كجزء من حالة التجميع، لذلك إذا قمت بأداء عملية تغيير اسم قاعدة بيانات، مثل ضغط قاعدة البيانات إلى اسم جديد، وتصبح قاعدة البيانات غير مجمعة.

تلميح يمكن تجنب إلغاء تجميع التطبيق عند ضغط التطبيق من خلال ضغط قاعدة البيانات لنفس الاسم.

أدوات تصحيح الأخطاء

تحتوي أدوات VBAعلى تعليمات تقوم بتحليل أخطاء الوقت الحالي التي تحدث عندما تكون VBA غير قادرة على تنفيذ جملة أثناء تشغيل الرمز والأخطاء الاعتبارية التي تحدث عند إمكانية قيام VBA بتنفيذ الرمز بدون إعطاء النتيجة المتوقعة.تحتوي VBA على نوعين من أدوات تصحيح الأخطاء:

- أدوات تتيح لك الاحتفاظ بما يحدث للبيانات أثناء تشغيل الرمز
- أدوات تقوم بتعليق تنفيذ الرمز بجملة وتتيح لك استخدام نافذة Immediate لاختبار وعرض البيانات

تحتوي الوحدات النمطية على ثلاثة حالات "تسمى أيضاً أطوار أو أوقات": وقت التصميم عندما تقوم بإدخال وتحرير الرمز، والمشغل عند تشغيل الرمز وطور الفصل عند تشغيل الرمز ولكنه يكون معلق بين جمل التنفيذ. معظم أدوات تصحيح الأخطاء تتطلب أن يكون الرمز في طور التوقف؛ وفي طور التوقف تقوم نافذة Module بعرض الرمز الذي يتم تشغيله والمتغيرات والخصائص التي تحتوي على قيم معلقة في وقت التنفيذ.

إدخال وإهمال طور التوقف

تقوم VBA بإدخال طور التوقف في خط رمز الحالات التالية:

- جملة في خط تقوم بتكوين خطاء أثناء التشغيل بدون أي مقاومة أخطاء عند حدوث الأخطاء
- وضع نقطة توقف على السطر. إذا قمت بإغلاق قاعدة البيانات ثم فتحها ثانية، سوف يتم مسح كل نقاط التوقف.
- وقوع جملة Stop على السطر التي يقوم بإيقاف التنفيذ ولكن يمكنك الاستمرار في تشغيل الرمز بواسطة اختيار Run ⇦ Continue.
- ضغط Ctrl+Break. اختيار Run ⇦ Break أو نقر زر Break أثناء تنفيذ الرمز.

II

يعتبر استخدام نقطة التوقف أو جملة Stop مفيد عند معرفة الجملة التي تسبب الخطأ أو منطقة الرمز العامة التي تحدث فيها الخطأ.



الاختلاف بين جملة Stop و end في الإجراء هو أن جملة End تقوم بإنهاء كل متغيرات مستوى الوحدة النمطية للاختبارات و التنفيذ وكذلك كل المتغيرات الثابتة في جميع الوحدات النمطية، بينما تقوم جملة بإيقاف جملة Stop بشكل مؤقت مع ترك القيم الحالية لكل المتغيرات بدون تأثير.

يسمى الإجراء الذي يتم تشغيله عندما يكون التنفيذ معلق وعندما يتم إدخال طور التوقف بالإجراء الحالي. تقوم VBA بعرض سهم أصفر في شريط الهامش يسار الجملة المتوقف عندها التنفيذ ويعرض النص بخلفية صفراء انظر الشكل ١٠-٥.وتسمى الجملة المعلمة بالجملة الحالية وهذه هي الجملة التي سوف يتم عرضها ثانية.



لترك طور التوقف و إعادة إدخال وقت تشغيل، يمكنك إما اختيار Run ⇦ Go/Continue أو نقر زر Continue في شريط الأدوات. لإنهاء تنفيذ الرمز وإعادة وضع كل المتغيرات، انقر زر Reset في شريط الأدوات.

عندما تقوم VBA بإدخال طور التوقف بواسطة تكوين معالجة خطأ أو خطأ مفصل ليس له تأثير، يقوم أكسس بعرض رسالة خطأ أثناء التشغيل. على سبيل المثال، إذا قمت بمحاولة تغيير خاصية Caption للنموذج Customers مستخدما جملة التخصيص:

Forms! Customers. Caption = "customer Information"

الشكل ١٠-٥

عندما يكوم الإجراء في طور التوقف، تقوم VBA بتعليم الجملة الحالية بسهم وخلفية ملونة

```
Form_Orders : Class Module
CustomerID    BeforeUpdate

Private Sub CustomerID_BeforeUpdate(Cancel As Integer)

    Dim strMsg As String, strTitle As String
    Dim intStyle As Integer

    If IsNull(Me!CustomerID) Or Me!CustomerID = "" Then
        strMsg = "You must pick a value from the Bill To list."
        strTitle = "Bill To Customer Required"
        intStyle = vbOKOnly
        MsgBox strMsg, intStyle, strTitle
        Cancel = True
    End If
```

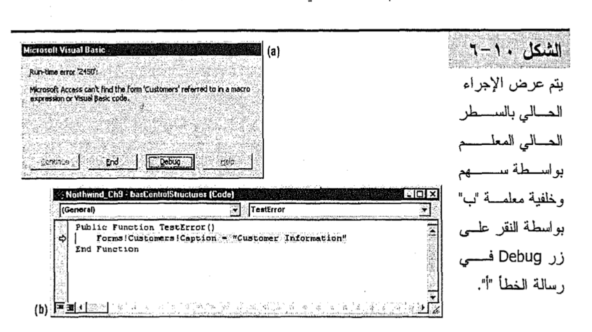
ولكن عندما يكون نموذج Customers غير مفتوح عند تشغيل الجملة، يتم تكوين خطأ عند التشغيل انظر الشكل ١٠-٦أ.

بالنقر على زر Debug في مربع حوار رسالة خطأ الإجراء، يتم عرض الإجراء الحالي في نافذة Module بالجملة الحالية المعلمة بواسطة السهم والخلفية الملونة انظر الشكل ١٠ .٦ب. يمكنك أحياناً تصحيح الخطأ مباشرة بواسطة تغيير الرمز. وفي هذا المثال، يمكنك تصحيح الخطأ بواسطة تضمين الرمز لاختبار ما إذا كان نموذج Customers مفتوح قبل تشغيل الجملة. وبالنقر على زر End في مربع رسالة خطأ المشغل يتم إنهاء تنفيذ الرمز، وعندما تنتقر هذا الزر، تحتفظ متغيرات المستوى حسب الوحدة بالقيم الخاصة بها. إذا كنت تريد إعادة وضع المتغيرات بعد نقر End، انقر زر Reset في شريط الأدوات.

التجربة في نمط التوقف

في حالة التنفيذ المعلق، يمكنك تنفيذ ما حدث قبل إيقاف الرمز بواسطة استخدام الأساليب التالية:

- التبديل إلى نوافذ أخرى بعد فحص حالتها
- فحص قيم المتغيرات والمراقبات والخصائص لمشاهدة ما إذا كانت بعض الجمل قامت بوضع قيمها خطاء.
- عمل تغييرات للرمز ثم الاستمرار في التنفيذ بواسطة اختيار Run ⇦ Continue.



الشكل ١٠-٦ يتم عرض الإجراء الحالي بالسطر الحالي المعلم بواسطة سهم وخلفية معلمة "ب" بواسطة النقر على زر Debug في رسالة الخطأ "أ".

إعادة تشغيل الرمز

تتطلب بعض التغييرات إعادة تشغيل الرمز. تقوم VBA بعرض رسالة عند الحاجة إلى ذلك.يمكنك إعادة تشغيل الرمز عند تشغيل البرنامج أو عندما يكون الرمز في طور التوقف.

بإعادة التشغيل يتم وضع كل المتغيرات بما فيها المتغيرات العامة والثابتة في قيمها الافتراضية الأولية: المتغيرات الرقمية تكون على صفر ومتغيرات الطول المتغير تكون على صفر والمتغيرات الثابتة على صفر ANSI، والمتغيرات المتغيرة على فارغ ومتغيرات الوحدة على لا شيء. ولإعادة تشغيل كل المتغيرات، اختر Run ⇦ Rest أو انقر زر Reset في شريط الأدوات.

وضع وإزالة نقاط التوقف

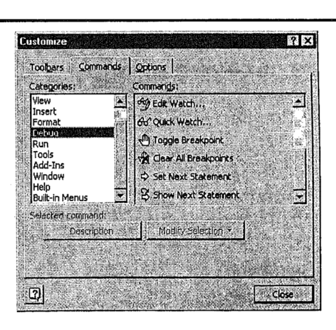
يمكنك وضع نقاط التوقف بواسطة النقر في شريط الهامش على يسار الجملة القابلة للتنفيذ، بواسطة وضع نقطة الإدخال في الجملة القابلة للتنفيذ ثم ضغط F9 أو بواسطة اختيار نقطة التوقفDebug ⇦ Toggle "كل من هذه الإجراءات يعتبر تبديل بواسطة أخذ الإجراء عندما تكون نقطة الإدخال في جملة نقطة التوقف، عليك إزالة نقطة التوقف."

عندما تقوم بوضع نقطة توقف، تقوم VBA بعرض دائرة حمراء في شريط الهامش ويعرض السطر ذات الخط الأبيض على خلفية حمراء افتراضياً. يمكنك وضع أكثر من نقطة توقف في الإجراء، ويمكنك وضع نقاط توقف في الإجراءات المكررة.

يمكنك إزالة كل نقاط التوقف في كل الوحدات النمطية بواسطة اختيار Debug<ClearAll Breakpointsأو بواسطة ضغطCtrl+Shift+F9. يمكنك تفصيل شريط أدوات محرر الفيجوال بيسك بواسطة إضافة زر Clear All Breakpoints. يوضح الشكل ١٠-٧ فئة Debug لأزرار شريط الأدوات بما فيها زر Clear All Breakpoints وأزرار لأوامر قوائم أخرى متعددة.

الشكل ١٠-٧

يمكنك تفصيل شريط محرر أدوات الفيجوال بيسك بواسطة إضافة زر Clear All Breakpoints.



عند تشغيل الإجراء بنقطة التوقف، تقوم VBA بتنفيذ جمل قبل نقطة التوقف ثم إدخال رمز التوقف بواسطة التوقف قبل تنفيذ جملة نقطة التوقف. وبحالة الإجراء في طور التوقف يمكنك التقدم بخطى منفردة من خلال الرمز أو أنه يمكنك ترك طور التوقف وتعود إلى طور التنفيذ العادي بواسطة نقر زر GO/Continue في شريط الأدوات أو بضغط F5 أو اختيار Run ⇦ Continue.

يمكنك استخدام نقاط التوقف للمساعدة في تحديد موضع الخطأ. عليك وضع نقطة توقف في الجملة التي تتوقع أنها تسبب المشكلة ثم قم بتشغيل الإجراء. وعندما تصل إلى نقطة التوقف ويكون الإجراء في حالة توقف، يمكنك تحديد ما إذا كان الخطأ قد حدث قبل الوصول إلى نقطة التوقف. وإذا كان الخطأ قد حدث بالفعل (في أي حالة، قد يقوم التنفيذ بالإنهاء قبل الوصول إلى نقطة التوقف)، سوف تعلم أن جملة سببت الخطأ قبل ذلك. يمكنك فحص الجمل السابقة ثم تقوم بوضع نقطة فصل مسبقة لاكتشافات أكثر. وإذا وصل الإجراء نقطة التوقف ولم يقع الخطأ بعد، يمكنك الاستمرار حتى تصل إلى الخطأ.

الاستمرار في الرمز

تحتوي VBAعلى ثلاثة طرق مختلفة تتيح لك الاستمرار في الجمل (تسمى هذه الطريق تتبع الجمل):

Step Into: إذا قمت بضغط F8 أو اختيار Debug<Step Into، سوف تقوم VBA بالتبديل إلى المشغل وتنفيذ الجملة الحالية ثم العودة ثانية إلى طور التوقف. ويعتبر مفتاح F8 واحد من أهم المفاتيح المفيدة في تصحيح أخطاء الإجراء. وعندما يكون السطر محتوي على أكثر من جملتين أو أكثر (مفصولتين بعلامة فاصلة) يمكنك الانتقال إلى الجملة الثانية في الوقت المحدد. وإذا كانت الجملة الحالية استدعاء لإجراء آخر، تواصل VBA إلى جمل الإجراء المستدعى بخطوة في الوقت المحدد، وعندما يقوم الإجراء المستدعى بالإنهاء، تعوم VBA إلى الجملة التالية في الإجراء الحالي.

Step Over: قد تحتاج أحياناً إلى الدخول إلى جمل في الإجراء، ولكن لن تكون في حاجة إلى الدخول في جمل إجراء تم استدعاءه. عند استخدام أمر Step Over والجملة الحالية تكون استدعاء للإجراء التالي، تقوم VBA بتنفيذ الإجراءات التي تم استدعاءها كخطوة فردية ثم الانتقال إلى الجملة التالية للإجراء الحالي. وعندما تستخدم أمر Step Over، تستمر نافذة Module في عرض الإجراء الحالي أثناء تشغيل الإجراء المستدعى. لاستخدام أمر Step Over، اختر Debug< Step Over أو اضغط Shift+F8.

Step Out: عند استخدام أمر Step Out وقد تم استدعاء الإجراء الحالي بواسطة إجراء آخر، تقوم VBA بتنفيذ المتبقي من الإجراءات الحالية التي تقوم بالاستدعاء بخطوة منفردة ثم تعود ثانية إلى الإجراء المستدعى ثم تتوقف، إذا لم يتم استدعاء الإجراء الحالي بواسطة إجراء آخر، يكون استخدام أمر Step Out متطابق تماماً مع استخدام أمر Continue. لاستخدام أمر Step Out، اختر Debug ⇦ Step Out أو اضغط Ctrl+Shift+F8.

يمكنك التبديل بين هذه الأنواع الثلاثة للانتقال. عندما تريد إيقاف تشغيل الإجراء بجملة في وقت محدد، اضغط F5 للاستمرار في التنفيذ بدون مقاطعة.

وضع الجملة التالية

إذا كنت في طور التوقف، يمكنك تخطي سطور أو الرجوع وتكرار الرمز بواسطة تحديد الجملة التالية التي تريد تنفيذها، استخدم واحدة من الطرق التالية:

- ♦ نقل نقطة الإدراج إلى أي سطر من الرمز في نفس الإجراء، انقر على زر الماوس الأيمن في السطر ثم اختر أمر جملة Set Next Statement في قائمة الاختصارات.
- ♦ نقل نقطة الإدراج إلى أي سطر من الرمز ثم اختر Debug ⇦ Set Next Statement.
- ♦ اضغط Ctrl+F9 بالمؤشر بوضعه على السطر الذي تريد استئناف تنفيذه.

تقوم VBA بتعليم الجملة المحددة مثل الجملة الحالية بالسهم الأصفر في الهامش والخلفية الصفراء. وبعد تحديد الجملة التالية المراد تنفيذها، يمكنك نقر زر GO/Continue في شريط الأدوات أو الأوامر Step Into أو Step Over في قائمة Debug.

يتيح لك أمر Debug ⇦ Show Next Statement مشاهدة الجملة التالية التي سوف تقوم بالتنفيذ في الإجراء.

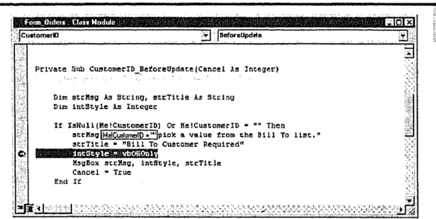
تلميح يمكنك إضافة أزرار شريط الأدوات إلى شريط الأدوات الخاص بأوامر Set Next Statement وShow Next Statement وStep over وStep Into وStep Out.

تنفيذ مجموعة من الأوامر

عندما يكون الإجراء في طور توقف، يمكنك استخدام أمر Run to Cursor لتشغيل البرنامج حتى يتم تشغيل السطر الذي تحديده. انقل نقطة الإدراج إلي أي سطر من الرمز في نفس الإجراء، ثم انقر بزر الماوس الأيمن واختر Run to Cursor أو اضغط Ctrl+F8 لتقوم VBA بتشغيل رمز يبدأ الجملة الحالية إلي السطر بنقطة الإدراج ثم أعد إدخال رمز التوقف. يتيح لك خيار Run to Cursor تجنب الدخول في كل جملة في الإجراء.

عرض القيم الحالية في نافذة Module

زود أكسس ٢٠٠٠ بخاصية تتيح لك عرض القيمة الحالية للمتغير أو الثابت المشار إليهما في الإجراء عندما كان الإجراء في طور التوقف. انقل مؤشر الماوس إلي المتغير أو الثابت الذي تريد فحصه. تقوم VBA بعرض القيمة الحالية مثل Data Tip مباشرة أسفل البند انظر الشكل ١٠-٨. يمكنك إغلاق هذه الخاصية بواسطة مسح مربع الاختيار Auto Data Tips بعلامة الجدولة Editor بمربع حوار خيارات Editor للفيجوال بيسك.



الشكل ١٠-٨
يمكنك فحص القيمة الحالية للمتغير أو الثابت المشار إليهما في الإجراء الموجود في طور التوقف.

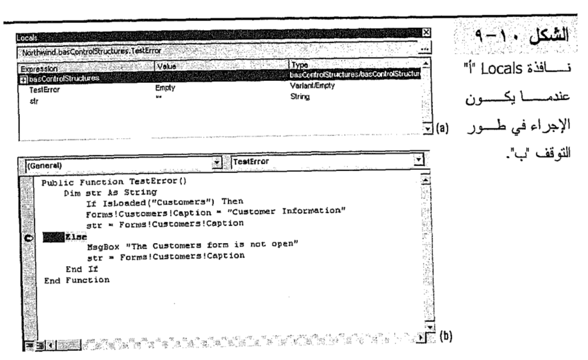
ملاحظة
كما علمت في الفصول السابقة، يمكنك استخدام نافذةImmediate لطباعة قيم حالية. تعتبر نافذة Immediate متاحة سواء كان الطور في طور التوقف أو لا.

استخدام نافذة Locals في طور التوقف

تعتبر نافذة Locals أداة ذات قيمة في تصحيح وحدات VBA النمطية. لفتح نافذة Locals، اختر Locals Window > View. تقوم Locals Window بوضع المتغيرات والثوابت المشار إليها

في الإجراء الحالي في عمود Expression وأنواع البيانات في عمود Type. يعرض عمود Value قيم المتغيرات والثوابت قبل أن تقوم الجملة الحالية بالتنفيذ. وعندما تنتقل في الجمل، تقوم Window Locals بالتحديث لعرض القيم الحالية. يمكنك أيضاً تخصيص قيم جديدة للمتغيرات في Locals Window. لتغيير قيمة، انقر القيمة في عمود Value ثم اضغط Enter.

عندما تكون VBA في طور التوقف، تقوم نافذة Module بعرض الإجراء الذي يتم تشغيله حالياً بالجملة الحالية المعلمة بواسطة السهم الأصفر وخلفية. يوضح الشكل ١٠-٩ أ نافذة Locals عندما يكون الإجراء في طور التوقف في الجملة الموضحة في الشكل ١٠-٩ ب.



الشكل ١٠-٩ نافذة Locals "أ" عندما يكون الإجراء في طور التوقف "ب".

تقوم نافذة Locals أيضاً بعرض معلومات الوحدات. عندما يكون الإجراء الحالي إجراء حدث مخزن في نموذج أو وحدة تقرير، تتضمن Locals Window معلومات حول الوحدات في التقرير أو النموذج المتعلق. يتم الرجوع إلى التقرير أو النموذج باسم Me. انقر على علامة Plus على يسار الوحدة في نافذة Locals لتوسيع الوحدة وعرض كل أجزائها.

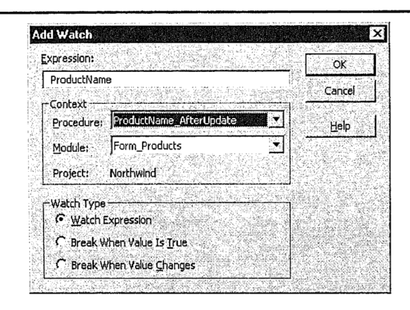
استخدام نافذة Watches

تتيح لك نافذة Watches عرض المتغيرات في أي إجراء وتقوم أيضاً بفحص قيم التعبيرات المفصلة التي تقوم بتكوينها. لفتح تعبيرات المشاهدة، اختر نافذة View ⇦ Wtach.

يمكنك تحديد تعبير مخصص أو تعبير تريد الاحتفاظ بقيمته أثناء تشغيل الرمز. تقوم بتحديد تعبيرات المشاهدة في مربع حوار Add Watch انظر الشكل ١٠-١٠ أ المتاح بواسطة اختيار

Debug ⇦ Add Watch . أدخل أي متغير أو خاصية أو استدعاء إجراء أو تعبير صحيح في مربع Expression. يمكنك تحديد السياق أثناء قيام الإجراء المخصص أو الوحدة النمطية التي يجب أن تكون حالية قبل VBA بتقييم التعبير أو يمكنك ترك السلسلة غير مقيدة والطلب من VBA بتقييم التعبير لكل الوحدات النمطية.استخدم خيار Watch Type لتحديد ما إذا كنت تريد عرض قيمة التعبير أو استخدام قيمته لتعليق تنفيذ الرمز أينما كانت قيمة التعبير True أو التوقف أينما كانت قيمة التعبير تغير القيمة.

بعد تحديد تعبيرات المشاهدة، قم بفتح نافذة Watches لعرض تعبيرات المشاهدة وقيمهم الحالية والسياق المحدد انظر الشكل ١٠-١٠ب. ولتحرير أو حذف تعبيرات المشاهدة، استخدم مربع حوار Edit Watch أو بواسطة ضغط Ctrl+W.



الشكل ١٠-١٠

تكوين تعبيرات Watch في مربع حوار Add Watch ومراجعة تغييرات التعبيرات في نافذة Watches "ب".

استخدام Quick Watch في طور التوقف

عندما تكون VBAفي طور التوقف، يمكنك كشف قيمة المتغير أو الخاصية أو استدعاء الوظيفة بدون تحديده أولاً كتعبير مشاهدة. لعرض قيمة التعبير في الإجراء الحالي، اختر التعبير في الإجراء ثم اختر Debug ⇦ Quick Watch أو اضغط Shift+F9. يقوم مربع رسالة Quick Watch بعرض التعبير وقيمته الحالية انظر الشكل ١٠-١١. يمكنك إضافة التعبير إلى نافذة Watches كتعبير مشاهدة بواسطة نقر زر Add.

Quick Watch

Context
[NWIND.MDB].Form_Orders.CustomerID_BeforeUpdate

Expression
strMsg

Add
Cancel

Value
"You must pick a value from the Bill To list."

الشكل ١٠-١١ استخدام مربع حوار Quick Watch لعرض قيمة المتغير أو الخاصية أو الوظيفة في الإجراء الحالي في طور التوقف.

تجنب الخطأ

أفضل طريقة للتعامل مع الأخطاء هي تجنبها من المرة الأولى إذا أمكن. تحدث معظم الأخطاء نتيجة تغييرات. كم مرة حدث لك لهذا؟ تقرر عمل تغيير صغير في التطبيق الذي يعمل جيدا ثم تجد أنه لا شيء يعمل جيدا بعد عمل التغيير. حتى معظم الإصلاحات الطفيفة قد تؤدي إلى استهلاك ساعات عدة لتصحيح أو إزالة كل الأخطاء التي نتجت عن التغيير. وهاك بعض التلميحات لتجنب الأخطاء:

التخطيط لتجنب الأخطاء: في مرحلة التخطيط للتطبيق، قد تكون عندك القدرة على التخطيط للتغييرات المستقبلية وكذلك تجنب الأخطاء التي تنتج عن هذه التغييرات. على سبيل المثال، عند تصميم الجداول، فكر في الحقول الإضافية التي قد تكون في حاجة إليها في المستقبل وتضمينهم في البداية. تأكد من العلاقات بين الجداول وهل هذه العلاقة واحد لمتعدد أم متعدد لمتعدد. على سبيل المثال، في تطبيق تعليمي يتتبع الطلاب، المعلمين والفصول؛ فالعلاقة العادية بين المعلمين والفصول علاقة واحد لمتعدد لأن الفصل له عادة معلم واحد وعلى العكس من ذلك إذا توقعت أن يقوم بالتعليم مجموعة من المعلمين فعليك تكوين علاقة متعدد لمتعدد بدل عن ذلك.

التدقيق في بناء الجملة: من الممكن أن يكون لديك مدقق VBA للصياغة أثناء الكتابة بواسطة خيار فحص Auto Syntax (في علامة الجدولة Editor لمربع الحوار Tools ⇦ Option في Editor للفيجوال بيسك). إذا كنت لا تريد التدقيق بهذا الخيار، فإن VBA تدقق الصياغة عندما تقوم بتجميع الرموز.

استخدام تعريف المتغير الظاهر: مصدر عام للأخطاء هو الأخطاء المطبعية في أسماء المتغيرات. يمكنك تجنب هذه الأخطاء بواسطة تضمين Option Explicit في قسم Option Explicit في كل الوحدات الجديدة إذا قمت بفحص خيار Require Variable Declaration (في علامة الجدولة Editor لمربع الحوار Tools ⇦ Option في Editor للفيجوال بيسك).

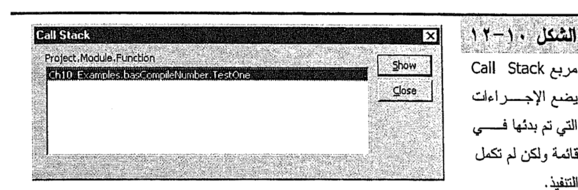
كتابة البيانات: يجب اختيار نوع البيانات لكل المتغيرات بوضوح. بواسطة تحديد المتغيرات بنوع بيانات محدد بدل من متغير أو أنواع بيانات وحدة، عليك تجنب أخطاء المشغل التي تنتج عندما تكون VBA غير قادرة على تحويل نوع البيانات للمتغير المتغير لكي يقوم بعمل حساب أو عندما تقوم VBA باكتشاف أن الرمز يشير إلى الخاصية أو الطريقة الغير ملائمة لمتغير الوحدة. وباستخدام أدق أنواع البيانات، تجبر VBA على تحليل دقة الإشارات أثناء وقت التجميع.

تجنب تداخل الأسماء: إذا قام إجراءان باستخدام متغير بنفس الاسم مثل متغير مستوى الوحدة، يحدث أن يقوم إجراء بالكتابة الفوقية على قيمة موضوعة بواسطة آخر خطاء، ويمكنك تجنب ذلك بواسطة تعريف المتغيرات بأصغر مساحة ممكنة. في حالة استخدام متغير في إجراء منفرد، عليك إذاً تعريف المتغير داخل الإجراء. وفي حالة استخدام المتغير بواسطة الإجراءات في وحدة منفردة، ثم قم بتعريف المتغير كمتغير تعريف مستوى وحدة خاص. وعلى وجه الخصوص، استخدم متغيرات تعريف مستوى وحدة عام فقط عندما تحتاج عمل المتغير متغير للإجراءات المتعددة في وحدات متعددة. قد تظهر أيضاً متناقضات التسمية إذا كان لديك إجراءات في وحدات منفصلة بنفس الاسم. وإذا قمت باستدعاء مثل هذا الإجراء من إجراء في وحدة أخرى، سوف يحدث خطأ ما لم تقم بتضمين اسم الوحدة في استدعاء الإجراء، لأن VBA غير قادرة على تحديد الإجراء الذي سوف يقوم باستدعائه.

استخدام التعليقات: أنت بحاجة لعقد موازنة معقولة بين كتابة التعليقات لكل جملة بدون أن تتضمن أي تعليقات في الرمز الخاص بك. والقاعدة الجيدة هي تضمين تعليقات كافية بحيث يمكن لمبرمج VBA فهم الرمز. وأنت ليس بحاجة إلى التعليق على كل جملة أو هيكل واضح ولكن قد تكون بحاجة لتضمين التعليقات في بداية كل وحدة وإجراء لبيان الغرض منه والإشارة إلى متغيرات محددة في الوحدة النمطية أو الإجراء. وبالمثل يجب تضمين تعليقات توضح العملية أن كل وحدة جمل تقوم بالتنفيذ. يأخذ كل تعليق جزء صغير من الآخر. ومع هذا تتكرر الأجزاء ومن الممكن أن تؤدي إلى بطأ في الرمز، لذلك من الحكمة تجنب التعليقات الزائدة والغير الضرورية.

استخدام مربع حوار Call Stack

عندما يحتوي الرمز على طبقات متعددة من الإجراءات المتداخلة، تعرف ما إذا كانت الإجراءات التي قمت باستدعائها بالشكل الذي تريده مساعدة أم لا. يمكنك استخدام هذا الحوار عندما تكون VBA في طور التوقف للتتبع من خلال عملية تطور الإجراءات المتداخلة. ولعرض هذا المربع، اختر مربع View< Call Stack أو اضغط Ctrl+L. يقوم هذا المربع بعرض قائمة بالإجراءات التي قمت ببدئها ولكن لم تتم التنفيذ انظر الشكل ١٠-١٢. وأول إجراء نشط في سلسلة الاستدعاء يكون أسفل القائمة مع الإجراءات التالية مضافة إلى القمة.



الشكل ١٠-١٢ مربع Call Stack يضع الإجراءات التي تم بدئها في قائمة ولكن لم تكمل التنفيذ.

معالجة الأخطاء

إن الهدف من الاختبار وتحري الأخطاء الناجح هو اكتشاف أكبر عدد ممكن من الأخطاء وتحليلها لمعرفة ما إذا كان من الممكن تصحيحها أو تجنبها بواسطة تغيير الرمز. حتى بعد إزالة الأخطاء، تبقى الأخطاء التي لا يمكن تجنبها بإهمال المستخدم إدخال جزئية المعلومات المطلوبة فتقوم الشبكة بالانقطاع بدون توقع أو حدوث شيء آخر. وكما تم بيانه في الفصول السابق فإنك تتعامل مع الأخطاء التي لا يمكن تجنبها بواسطة كتابة رمز VBA لاستبدال معالجة الأخطاء الافتراضية برمز معالجة الأخطاء المفصل. وبطريقة أخرى حتى لو لم يمكنك تجنب خطأ فإنه يمكنك توقعه.

الأخطاء القابلة للحصر

يوجد ثلاثة حالات لأكسس يمكنها أن تسبب أخطاء:

- يقوم محرك قاعدة البيانات بمعالجة الأخطاء التي تحدث عندما لا يمكنها تنفيذ مهمة تتطلب أي من وحدات البيانات.

- يقوم وسيط أكسس بمعالجة أخطاء الماكرو وأخطاء أخرى متعلقة بالوسيط ذاته.
- تقوم VBA بمعالجة أخطاء المشغل بواسطة VBA ذاتها ورمز VBA الخاص بك.

عندما خطأ يحدث نتيجة بسبب عدم القدرة على تنفيذ مهمة ما، يقوم تطبيق أكسس أو محرك قاعدة البيانات أو VBA بتحديد الخطأ من قائمة الأخطاء المحددة مسبقاً وتحديد رمز الخطأ الرقمي للخطأ. يمكنك كتابة رمز معالج الخطأ للأخطاء التي تم تخصيصها برقم فقط، وتسمى هذه الأخطاء بالأخطاء القابلة للحصر.

يحتوي كل من محرك قاعدة البيانات وVBA على مجموعة رموز الأخطاء الخاص بها وطريقتها في معالجة معلومات الأخطاء بينما يقوما كل من جيت وMSDE بتخزين معلومات الخطأ في مجمع الأخطاء. تقوم VBA باستخدام وحدة الأخطاء الخاصة بها وفي كلتا الحالتين يتم إعادة كتابة الوحدة في كل مرة يحدث فيها خطأ جديد. تقوم المعلومات في مجمع الأخطاء ببيان خطأ VBA الأخير، وعند حدوث خطأ جيت أو MSDE. وVBA يتم مسح وحدة الأخطاء أو مجمع الأخطاء وإعطاء معلومات حول الخطأ الجديد.

ملاحظة عندما يكون الوسيط أو محرك قاعدة البيانات غير قادر على تنفيذ مهمة، توجد إمكانية حدث منفرد يسبب أكثر من خطأ. فعلى سبيل المثال غالباً ما تحتوي الأخطاء المتعلقة بقواعد بيانات ODBC على مجموعة من الأخطاء المتعلقة بخطأ مختلف لكل مستوى من مستويات محركات ODBC. يقوم محرك قاعدة البيانات بتخزين كل خطأ متعلق في وحدة أخطاء منفصلة؛ ومجموعة الأخطاء المكونة بواسطة حدث منفرد يقوم بتجميع مجمع الأخطاء. كذلك يقوم مجمع الأخطاء بتخزين التفاصيل حول كل الأخطاء المتعلقة والتي تم التعرف عليها عند حدوث خطأ الوصول إلى البيانات.

رسائل الأخطاء المفصلة

لابد أنك معتاد على رسائل الأخطاء الافتراضية التي تحدث عند العمل مع أكسس:

Index or primary key can't contain a null value.
Duplicate value in index, primary key or relationship. Changes were _ unsuccessful.
The text you enter must match an entry in the list.

رسائل مثل هذه لها معنى قليل للمستخدم المبتدئ للتطبيق. ومن أسباب معالجة الأخطاء بنفسك هو استبدال رسالة الخطأ الافتراضية بالرسالة المفصلة التي هي أكثر معلومات ومساعدة.

أخطاء محرك قاعدة البيانات والوسيط

عند العمل في نموذج أو تقرير، من الممكن أن تقوم بعمل بعض الأخطاء في محرك قاعدة البيانات ووسيط أكسس. على سبيل المثال، عندما تقوم بإدخال قيمة في المربع القابل للتحرير الغير موجودة في القائمة القابلة للتحرير ووضع Limit to List على Yes، لا يتمكن أكسس من قبول قيمة وتكوين خطأ وسيط في رمز الخطأ ٢٢٣٧، يقوم أكسس بعرض رسالة الخطأ الافتراضية لهذا الرمز وإلغاء تحديث المربع القابل للتحديث. عندما تحاول حفظ السجل وتكوين بيانات للوصول إلى الخطأ برمز الخطأ ٣٠٥٨، يقوم أكسس بعرض رسالة الخطأ الافتراضية لهذا الرمز وإلغاء عملية الحفظ.

استخدام حدث الخطأ في النموذج ووحدات التقرير

عند تكوين خطأ محرك أو خطأ وسيط، يتمكن النموذج أو التقرير من التعرف على حدث الخطأ. يمكنك تكوين إجراء خطأ خاص بحدث الخطأ لمعالجة الخطأ بواسطة قطع الاستجابة الافتراضية واستبدال رسالة الخطأ الافتراضية برسالة الخطأ المفصلة وتحديد الإجراءات التي تريد أكسس أن يأخذها بدل من السلوك الافتراضي. على سبيل المثال، يمكنك معالجة المفتاح الأساسي الفارغ "رمز الخطأ = ٣٠٥٨" بواسطة عرض رسالة مفصلة وإخفاء رسالة الافتراضية ونق التركيز إلى مراقبة المفتاح الأساسي وإلغاء عملية الحفظ.

يحتوي قالب رمز الإجراء لحدث الخطأ المعروف بواسطة النموذج على الصيغة:

Index or primary key can't contain a null value.

Duplicate value in index, primary key or relationship. Changes were _
unsuccessful.

The text you enter must match an entry in the list.

تقوم باستخدام المعاملات للوصول إلى أكسس ويعتبر معامل DataErr رمز خطأ الوسيط أو خطأ المحرك الذي حدث. عند حدوث خطأ محرك أو وسيط، يقوم أكسس باجتياز رمز الخطأ كقيمة DataErr لإجراء حدث خطأ التقرير أو النموذج وتقوم باستخدام معامل Response لإخبار أكسس بعرض أو إخفاء رسالة الخطأ الافتراضية. يمكنك استخدام ثابت للمعامل Response: DataErrContinue لإخفاء رسالة الأخطاء الافتراضية أو ac DataErrDisplay لعرضها.

لاكتشاف هذه المفاهيم، سوف تقوم بتكوين معالج خطأ يقوم بمعالجة أخطاء محددة في نموذج الموظفين في قاعدة البيانات Northwind. والخطوة الأولى هي تحديد رموز الخطأ التي معالجتها. وطريقة إجراء ذلك هي تكوين إجراء حدث بسيط لعرض رمز الخطأ ثم إطلاق الخطأ مثل التالي:

```
Private Sub Form_Error(DataErr As Integer, Response As Integer)
```

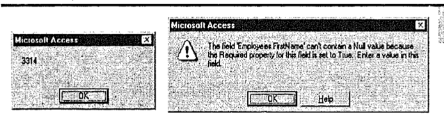
عندما يتعرف النموذج على حدث الخطأ، تقوم جملة MsgBox بعرض رمز الخطأ. عندما تقوم بنقر OK لهروب الرسالة، يقوم أكسس بعرض رسالة الخطأ الافتراضية المتعلقة برمز الخطأ.

اتبع الخطوات التالية لإدخال واختبار الإجراء:

١- استيراد نموذج Employees وجدول Employees إلى قاعدة بيانات CH10_Examples من قاعدة بيانات عينة Northwind.

٢- فتح نموذج Employees في عرض Design. انقر خاصية OnError، انقر زر Build ثم أدخل إجراء حدث Form_Error الموضح أعلاه.

٣- حفظ النموذج والتبديل إلى عرض Form. عرض سجل جديد وإدخال الاسم الأخير بدون إدخال الاسم الأول وضغط Shift +Enter لحفظ السجل. يقوم مربع الرسالة بعرض رقم الخطأ ٣٣١٤ متبوع برسالة الخطأ الافتراضية انظر الشكل ١٠-١٣. يقوم محرك جيت بتكوين هذا الخطأ لأن حقل FirstName هو حقل Required.

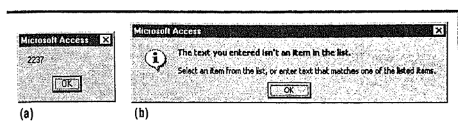


الشكل ١٠-١٣

يقوم إجراء حدث Form_Error بعرض رمز الخطأ "أ" ويقوم أكسس بعرض رسالة الخطأ الافتراضية للخطأ الذي تم توليده بواسطة محرك قاعدة البيانات جيت "ب".

٤- إدخال الاسم الأول وكذلك الاسم الأخير والانتقال إلى المربع القابل للتحرير Reports To وإدخال zzzz. عندما تحاول الخروج من المربع القابل للتحرير، يقوم مربع الرسالة بعرض رمز الخطأ ٢٢٣٧ متبوعة برسالة الخطأ الافتراضية انظر الشكل ١٠-١٤.

٥- اختر اسم من القائمة القابلة للتحرير Reports To. اضغط Shift +Enter لحفظ السجل. يمكنك الآن تعديل إجراء الحدث لمعالجة هذه الأخطاء الثلاثة.



الشكل ١٠-١٤
يقوم إجراء الحدث Form_Error بعرض رمز الخطأ "أ" ويقوم أكسس بعرض رسالة الخطأ الافتراضية عند اختيار بند غير موجود في قائمة المربع القابلة للتحرير "ب".

يمكنك استخدام إما هيكل If...Then...else أو هيكل Select Case لعمل حصر لكل خطأ محدد. يقوم الإجراء الموضح أدناه باستخدام هيكل Select Case لتجربة قيمة رمز الخطأ ومعالجة الأخطاء.

```
Private Sub Form_Error(DataErr As Integer, Response As Integer)
    Select Case DataErr
        Case 2237
            MsgBox "You have made an invalid entry. Click the " _
                & "list to display the valid choices."
            Response = acDataErrContinue
        Case 3314
            MsgBox "You must enter both first and last "_
                & "names before you can save the record."
            Response = acDataErrContinue
        Case Else
            Response = acDataErrDisplay
    End Select
End Sub
```

في حالة حدوث واحد من هذه الأخطاء، يقوم الإجراء بعرض رسالة مفصلة ووضع Response to acDataErrContinue بحيث لا يتمكن أكسس من عرض رسالة الخطأ الافتراضية. وفي حالة عدم حدوث أخطاء محصورة، ولكن خطأ آخر يحدث، فإنه يتم تنفيذ

Case Else البديل ووضع Response to acDataErrDisplay يقوم بعرض رسالة الخطأ الافتراضية للخطأ الذي يمكن حصره. والهدف من Case Else البديل هو تحديد نوع الإجراء الذي عليك اتخاذه في حالة حدوث حدث غير متوقع.

قم بتعديل إجراء الحدث Form_Error الموضح أعلاه ثم قم بتكرار الخطوات الثلاث من خلال خمسة من التطبيق السابق لتجريبه.

يتم تصميم إجراء الحدث Form_Error للتعامل مع الوسيط وأخطاء جيت التي تحدث عندما يحتوي النموذج على مراقبة، لا يمكنك استخدام هذا الإجراء لحصر أخطاء VBA. وفي حالة حدوث أخطاء VBA في حالة تشغيل إجراء الحدث Form_Error، فإنه يتم عرض رسالة خطأ مشغل VBA. عليك التعامل مع أخطاء مشغل VBA بواسطة كتابة رمز معالجة الخطأ داخل الإجراء. يوضح القسم التالي كيفية معالجة أخطاء VBA.

أخطاء VBA

عند حدوث خطأ VBA عند تشغيل الإجراء، يكون معالج الخطأ الافتراضي أن VBA تعرض رسالة خطأ وتعلق تنفيذ الرمز في السطر الذي لم يتم تنفيذه. ولقد تم تنفيذ إجراء الحدث بواسطة Command Button Wizard وتزويد نموذج لمعالج خطأ VBA.

```
Private Sub cmdReturn_Click()
    On Error GoTo Err_cmdReturn_Click
        DoCmd.Close
Exit_cmdReturn_Click:
Exit Sub
Err_cmdReturn_Click:
   MsgBox Err.Description
   Resume Exit_cmdReturn_Click
End Sub
```

بواسطة تفاصيل برمجية حالية، تتبع معالجي أخطاء VBA مفاهيم برمجية قديمة: تستخدم إجراءات معالجة الأخطاء *علامات السطر* لتعريف سطور الرمز واستخدام جمل لمواجهة تدفق التنفيذ إلى السطور المعلمة. لابد أن تبدأ علامة السطر في العمود الأول وأن تبدأ بحرف وتنتهي بعلامة الترقيم (:).

بناء معالج أخطاء VBA

تستخدم معالجي الأخطاء البسيطة التي يقوم Command Button Wizard بتكوينها النموذج التالي:

```
'enable custom error-handling
On Error GoTo Err_procedurename
  [statements]
'begin exit code
Exit_procedurename:
  [statements]
Exit Sub
'begin error-handler
Err_procedurename:
  [error-handling statements]
  Resume Exit_procedurename
End Sub
```

تمكين معالج الخطأ تقوم باستخدام جملة On Error لإخبار VBA أنك سوف تقوم بمعالجة الخطأ بنفسك. ضع جملة On Error في بداية الإجراء بعد جمل التعريف الثابت والمتغير وقبل أي جمل قابلة للتنفيذ. تحتوي جملة On Error على النماذج الثلاثة التالية:

On Error Go To linelabel: تقوم بتحديد موضع معالج الخطأ داخل الإجراء. وعند حدوث خطأ، ينتقل التحكم إلى معالج الخطأ.

On Error Resume Next: تخبر أكسس بتجاهل الخطأ والاستمرار في الجملة باتباع الجملة التي سببت الخطأ مباشرة

On Error GoTo: تعطيل أي معالج خطأ تم تمكينه في الإجراء. تقوم هذه الجملة بإعادة وضع قيمة وحدة Error.

بواسطة تضمين جملة On Error، تقوم بتمكين معالج الخطأ المفصل ووضع حصر خطأ. تتبع جمل الإجراء جملة On Error. بينما تتبع الجملة Exit Sub أو Exit Function أو Exit Property الجملة الأخيرة للإجراء. ويت وضع رمز معالج الخطأ بشكل عادي بعد جمل الإجراء، لذلك تحتاج إلى لجملة Exit لمنع رمز معالج الخطأ من التشغيل عند حدوث الخطأ. تقوم جملة On Error بتحديد المكان الذي تنتقل إليه VBA في حالة حدوث أي خطأ في الجمل التي تم إدراجها بعد جملة On Error وقبل جملة Exit.

ملاحظة لتجنب أخطاء الصياغة، لاحظ أنه عند كتابة إجراءات معالج الخطأ في VBA، يمكنك تكوين جمل On Error "كلمتان". وعند تخصيص إجراء لنموذج في حدث يتم وقوعه، تقوم بوضع خاصية On Error "كلمة واحدة".

تنفيذ رمز معالج الخطأ: يقوم السطر ذات العلامة بتعليم بداية رمز معالجة الخطأ. يجب أن يكون هذا السطر في نفس الإجراء مثل جملة On Error. وفي حالة حدوث ينتقل التحكم إلى رمز معالجة الخطأ. يسمى معالج الخطأ الموجود في عملية معالجة الخطأ معالج الخطأ النشط. عند حدوث، عند تشغيل إجراء VBA، يتم تخزين المعلومات عن الخطأ كإعدادات الخاصية لوحدة VBA Err (يقوم الفصل الخامس ببيان الخصائص والطرق لوحدة ERR).أكثر الخصائص المستخدمة هي خاصية Number التي تقوم بإعادة رمز الخطأ كعدد Long و Description صحيح الذي يقوم بإعادة مسلسل كبيان خطأ.

جملة عامة في رمز معالجة الأخطاء هي عرض مربع الرسالة برمز وبيان الخطأ كالآتي:

```
MsgBox "Error number: " & Err.Number & " - " & Err.Description
```

من الممكن أن يقوم رمز معالجة بتجربة قيمة Err.Number واستخدام إما هيكل If...Then...else أو هيكل Select Case لتزويد الجمل البديلة لكل من الأخطاء المخصصة التي يقوم الرمز بمعالجتها. تقوم مجموعة الجمل للخطأ بتحديد الإجراءات التي عليك اتخاذها في حالة حدوث خطأ بهذا الرقم. اكتب رمز معالجة الأخطاء لكل الأخطاء التي تتوقعها وتتضمن رمز معالجة الخطأ للخطاء التي لا يمكن توقعها بالمثل.

على سبيل المثال، عندما تكون قد قمت بتعريف رموز الخطأ لأخطاء متعددة تريد حصرها في إجراء، استخدام هيكل Select Case في معالج الأخطاء لتزويد مجموعات بديلة من الجمل لكل خطأ واستخدام هيكل Select Else لحصر الأخطاء الغير معروفة والغير متوقعة.

```
Err_procedurename:
    Select Case Err.Number
    Case 3058
        [statements to handle the 3058 error]
    Case 3022
        [statements to handle the 3022 error]
    Case Else
        MsgBox "Error number: " & Err.Number & " - " & Err.Description
    End Select
Resume Exit_procedurename
```

خروج رمز معالج الخطأ: عند انتهاء رمز معالجة الخطأ، تحتاج إلى تحديد ما الذي على VBA القيام به بعد ذلك. وهناك ثلاث جمل عامة لخروج رمز معالج الخطأ:

Resume: تقوم باستئناف تنفيذ البرنامج بدءا بالجملة التي سببت الخطأ.

Resume Next: تقوم باستئناف تنفيذ البرنامج بدءا بالجملة مباشرة بعد الجملة التي سببت الخطأ. استخدم Resume Next عندما تريد الاستمرار في تشغيل الإجراء بدون إعداد محاولة الجملة التي سببت الخطأ.

Resume line: تقوم باستئناف التنفيذ في العلامة التي تم تحديدها بواسطة Line|. يجب أن تكون علامة السطر في نفس الإجراء ممثل معالج الخطأ. استخدم جملة Resume line عندما تريد الانتقال إلى الجملة قبل جملة خروج الإجراء.

تكوين معالج أخطاء عام

لك الإجراءات والبسيطة على وجه الخصوص، يجب أن تقوم بتضمين معالج أخطاء عام واحد على الأقل الذي يقوم بعرض رقم وبيان الخطأ لأي خطأ قابل للحصر يمكن حدوثه. يوضح الرمز التالي الجمل لكل معالج أخطاء عام والذي يتضمن هيكل Select Case للتجهيز للأخطاء المخصصة بالحصر.

```
On Error GoTo Err_procedurename
[statements]
Exit_procedurename:
Exit Sub
Err_procedurename:
   Select Case Err.Number
   Case Else
      MsgBox "Error number: " & Err.Number & " - " & Err.Description
   End Select
Resume Exit_procedurename
```

كمثال بسيط، يقوم إجراء Division بقسمة رقم على الآخر.

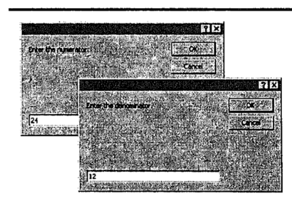
```
Public Sub Division()
   Dim dblnum As Double, dblden As Double, dblResult As Double
   On Error GoTo Err_Division
   dblnum = InputBox("Enter the numerator.")
   dblden = InputBox("Enter the denominator.")
```

```
      dblResult = dblnum / dblden
      MsgBox "The quotient is " & dblResult
   Exit_Division:
   Exit Sub
   Err_Division:
      Select Case Err.Number
      Case Else
         MsgBox "Error number: " & Err.Number & " - " & Err.Description
      End Select
      Resume Exit_Division
   End Sub
```

اتبع هذه الخطوات لتكوين وتجربة إجراء:

١- تكوين وحدة جديدة تسمى basErrors في قاعدة بيانات انظر الشكل ١٠-١٥ أ وإدخال الإجراء Division في الوحدة basErrors.

٢- تشغيل إجراء Divisionفي النافذة Immediate. أدخل ٢٤و١٢ في مربعات النص انظر الشكل ١٠-١٥أ. يقوم الإجراء بعرض الحاصل انظر الشكل ١٠-١٥ب.

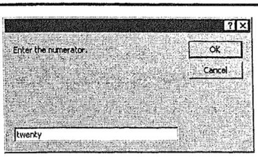


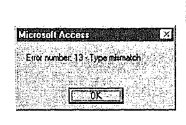


الشكل ١٠-١٥

يقوم الإجراء Division بجمع رقمين "أ" ويعوض الحاصل "ب".

٣- قم بتشغيل الإجراء ثانية. أدخل Twenty في مربع المدخلات الأول انظر الشكل ١٠-١٦أ. سوف يحدث خطأ لأن الإجراء يقوم بتوضيح db inum كرقم من نوع البيانات Double، ولا يتمكن أكسس من وضع هذا المتغير للمسلسل الذي قمت بإدخاله. يكون معالج الخطأ نشط وتدل رسالة خطأ العمود أن الخطأ يحتوي رمز ١٣ لنوع لا يطابق الخطأ انظر الشكل ١٠-١٦ب.





الشكل ١٠-١٦ عندما تقوم بإدخال مسلسل بدل من رقم "أ"، تقوم VBA بتكوين خطأ بالرمز ١٣ "ب".

٤- قم بتشغيل الرمز ثانية. وفي هذه المرة قم بإدخال ٢٠ في مربع المدخلات الأول و"صفر" في مربع المدخلات الثاني. يحدث خطأ آخر لأن أكسس لا يتمكن من قسمة الرقم على صفر. يعتبر معالج الخطأ نشط ويدل على أن الخطأ به رمز ١١ للقسمة على خطأ الصفر.

يمكنك الحصر على وجه الخصوص لكل واحد مكن هذه الأخطاء بواسطة استخدام مربع الرسالة لإبلاغ المستخدم أن الرقم لابد أن يتم إدخاله عند حدوث الخطأ برمز ١٣ ومربع رسالة منفصل يخبر المستخدم أن الرقم الذي تم إدخاله للمسمي لابد أن يكون صفر. يوضح إجراء Division الموضح أدناه هذا النوع من معالجة الأخطاء.

```
Public Sub Division()
   Dim dblnum As Double, dblden As Double, dblResult As Double
   Dim strAnswer As String
   On Error GoTo Err_Division
   dblnum = InputBox("Enter the numerator.")
Err_Denominator:
   dblden = InputBox("Enter the denominator.")
   dblResult = dblnum / dblden
   MsgBox "The quotient is " & dblResult
Exit_Division:
Exit Sub
Err_Division:
   Select Case Err.Number
   Case 13
      strAnswer = MsgBox("You must enter a number. Do you want " _
         & "to try again?", vbYesNo)
      If strAnswer = vbYes Then Resume
   Case 11
```

```
      strAnswer = MsgBox("You must enter a non-zero number. Do " _
        & "you want to try again?", vbYesNo)
      If strAnswer = vbYes Then Err_Denominator:
    Case Else
      MsgBox "Error number: " & Err.Number & " - " & Err.Description
    End Select
    Resume Exit_Division
  End Sub
```

تستخدم هذه النسخة وظيفة MsgBox لتتيح للمستخدمين اختيار محاولة استخدام الإجراء ثانية من عدمه، يقوم المتغير strAnswer بإعاقة اختيار المستخدم. عند حدوث أي خطأ ويقوم المستخدم باختيار المحاولة ثانية، يقوم الإجراء باستخدام جملة Resume لإبلاغ أكسس بالعودة إلى الجملة التي سببت الخطأ كالآتي:

```
  If strAnswer = vbYes Then Resume
```

إذا قام المستخدم باختيار عدم المحاولة ثانية، ينتقل التحكم إلى الجملة الأخيرة لمعالج الخطأ وينتقل التحكم آنذاك إلى رمز الخروج.

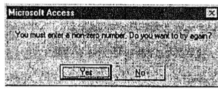
```
  Resume Exit_Division
```

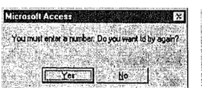
يستمر الإجراء في تضمين Case Else بديل لحصر الأخطاء غير المتوقعة.

قم بتعديل إجراء Division كما هو موضح أعلاه وقم بتشغيله في إطار Immediate. اكتب Twenty في مربع النص لمربع المد المدخلات الأول ثم انقر OK. وفي هذه المرة يقوم الإجراء بالحصر الأخطاء وعرض الرسالة الموضحة في الشكل "١٠-١٧ أ". إذا قمت بكتابة رقم في مربع النص لمربع المدخلات الأول وصفر في مربع النص الثاني، يقوم الإجراء بحصر الخطأ وعرض الرسالة الموضحة في الشكل "١٠-١٧ ب".

الشكل ١٠-١٧

تعرض تعليمات معالجة الخطأ المفصلة مربعات الرسالة هذه بحيث يتمكن المستخدم من تصحيح الخطأ والمحاولة ثانية.







تقوم Resume بإرسال إجراء VBA ثانية إلى السطر الذي سبب الخطأ أولاً. كذلك قد تحتاج إلى وضع علامة للسطر الذي يسبق هذا السطر الذي سبب الخطأ لكي يجبر المستخدم على أخذ إجراء ما آخر كما هو موضح في المثال الماضي عندما يقوم المستخدم بوضع صفر مثل المقام. وخلاف ذلك يقوم الإجراء بعرض الرسالة المفصلة ثم أنه عندما يقوم المستخدم بنقر Yes، تقوم VBA بالإعادة إلى سطر القسمة بدون إعطاء المستخدم فرصة إعادة إدخال مقام آخر.

وكبديل عن الانتقال من جزء من الإجراء إلى آخر، يمكنك تضمين الجمل Exit Sub في أي وقت تريد إنهاء البرنامج فيه، كما هو موضح في التقسيم التالي لإجراء Division. قد يكون هذا الأسلوب أسهل في القراءة على الرغم أن بعض المبرمجين يفضلون تزويد نقطة خروج واحدة فقط للإجراء.

```
Public Sub Division()
   Dim dblnum As Double, dblden As Double, dblResult As Double
   Dim strAnswer As String
   On Error GoTo Err_Division
   dblnum = InputBox("Enter the numerator.")
Err_Denominator:
   dblden = InputBox("Enter the denominator.")
   dblResult = dblnum / dblden
   MsgBox "The quotient is " & dblResult
Err_Division:
   Select Case Err.Number
   Case 13
      strAnswer = MsgBox ("You must enter a number. Do you want " _
         & "to try again?", vbYesNo)
      If strAnswer = vbYes Then Resume Else Exit Sub
   Case 11
      strAnswer = MsgBox ("You must enter a non-zero number. Do " _
         & "you want to try again?", vbYesNo)
      If strAnswer = vbYes Then Resume Err_Denominator Else Exit Sub
   Case Else
      MsgBox "Error number: " & Err.Number & " - " & Err.Description
   End Select
End Sub
```

الأخطاء في الإجراءات المستدعاة

عند حدوث خطأ في الإجراء المستدعى، تقوم VBA بالبحث عم معالج الخطأ أولاً في الإجراء المستدعى الذي يتم تشغيله عند حدوث الخطأ.

وفي حالة عجزه في البحث عن خطأ المعالج في الإجراء المستدعى، تقوم VBA بالاسترجاع إلى إجراء الاستدعاء وتنفيذ معالج الخطأ الخاص به إذا كان إجراء الاستدعاء يحتوي على معالج. وفي حالة مواجهته لسلسة من إجراءات الاستدعاء، يقوم أكسس بالتتبع الخلفي من خلال سلسلة الاستدعاء وتنفيذ معالج الخطأ الأول الذي يجده.

وإذا لم يمكنه العثور على معالجين خطأ، يقوم أكسس بتنفيذ البرنامج الفرعي لمعالجة الخطأ الفرعي. في هذه الحالة، يقوم أكسس بعرض رسالة الخطأ الافتراضية أولاً ثم عرض مربع حوار خطأ المشغل، ثم يقوم بإنهاء الإجراء.

تلميح للمزيد من المعلومات حول التعامل مع الأخطاء المستدعاة، انظر كتيب مطور أكسس ٢٠٠٠ بواسطة litwin Paul وGetz ken وmikeGilbert "sybex,1999".

خلاصة

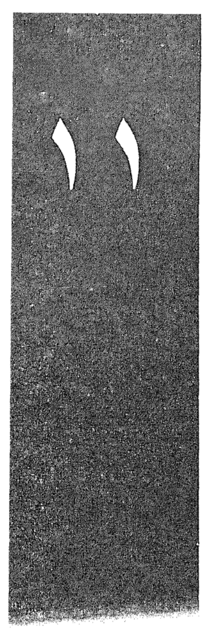
لقد عرض هذا الفصل الوحدات المتطورة لمعالجة الأخطاء. ولقد تم تغطية النقاط الهامة التالية:

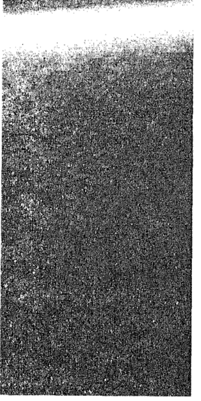
- نوعان من الأخطاء يحدثان في البرامج: يمكن تجنبه والذي لا يمكن تجنبه. والتصميم والتجريب وتحري الأخطاء الدقيق من المكن أن يحذف العديد من الأخطاء التي يمكن تجنبها.

يمكنك استخدام رمز معالجة الأخطاء المفصلة لتحديد الإجراءات التي تريد اتخاذها عند حدوث خطأ لا يمكن تجنبه.

- تزود VBA بفحص صياغة مضمن وتجميع البرامج الفرعية لإزالة أخطاء الترجمة.
- يزود أكسس بنوعين من أدوات تحري الأخطاء: تلك التي لا تقطع التنفيذ وتلك التي تستخدم التنفيذ المعلق ووضع الإجراء في طور التوقف.
- في حالة تعليق التنفيذ، يمكنك تنفيذ جملة مفردة في وقت محدد وتجريب تأثيره. يمكنك تطبيق قيم المتغيرات وعرض قائمة مرتبة بالإجراءات المستدعاة.

- يمكنك استخدام حدث Error لنموذج أو تقرير لحصر وكتابة رمز معالجة الخطأ لأخطاء الوسيط ومحرك قاعدة البيانات. ويعتبر حدث Error مفيد لاستبدال رسالة الخطأ الافتراضية برسائل الخطأ المفصلة.
- يمكنك استخدام جملة ON Error في إجزاء لتمكين رمز معالجة الأخطاء لأخطاء VBA التي تحدث عند تشغيل الإجراء. الغرض الرئيسي من معالجة الأخطاء المفصلة هو منع مربع خطأ المشغل وفشل الإجراء.
- يستخدم رمز معالجة الخطأ خصائص وحدة VBA Err لتحديد رمز الخطأ.
- يجب أن يكون لكل الإجراءات والأبسط من على وجه الخصوص معالج خطأ بسيط يقوم بعرض رمز الخطأ ورسالة الخطأ الافتراضية. من الممكن أن يقوم معالج الخطأ بإخراج الإجراء بدون تنفيذ جملة المشكلة أو تضمين التعليمات لأكسس لمتابعة حدوث الخطأ.





التنقل بواسطة VBA

أكسس

لقد وضعت الفصول الخمس السابقة أساسيات VBA. وتعد تلك الفصول الخمسة مرجـع لبـاقي الكتاب. فبواسطة الأدوات الأساسية والمهارات المناسبة نحن مستعدون أن نضع المفـاهيم معـاً ونقوم بإنشاء الإجراءات اللازمة في تطبيق قاعدة البيانات.

يركز هذا الفصل على التنقل. حيث يتناول الجزء الأول التنقل الذي يستخدم النمـاذج ويقـوم بتقديمك إلى بعض القوى الحقيقية التي تخص Access VBA. يفسر الفصل كيف تستخدم خاصية Recordset Clone للتنقل in memory من خلال سجلات النماذج المستقلة عن التنقل in the interface الذي يوفره النموذج. ستتعلم كيف تجعل التطبيقات سهلة الاستخدام عن طريق توفـير أزرار مخصصة على النماذج حتى يستطيع المستخدم أن ينتقل بسهولة بين النمـاذج، وعنـاصر التحكم، والسجلات باستخدام الأزرار المخصصة كمرشدين. وستتعلم أيضاً كيف تجعل التنقل آليـاً للسجلات في النماذج المستندة على المكان الطبيعي لهم، وكيف تجعل البحث عن سـجل يحقـق مقاييس البحث آلياً.

ويوضح الجزء الثاني للفصل طريق جديد للعمل مع السجلات. يمكن باستخدام VBA أن تفتـح مجموعة من السجلات في الذاكرة بدون فتح نموذج. يعد العمل مع سجلات في الذاكـرة أسـرع بكثير من العمل مع السجلات في النماذج، لأن Access لا تحتاج أن تستغل الوقت لتحمل نموذج داخل الذاكرة ولتقوم بإنشاء التمثيل المرئي الخاص بهم على الشاشة.

تلميح إحدى أساليب تعلم كيفية استخدام VBA هو إنشـاء مسـودة لتطبيـق مـا باستخدام Database Wizard وبعد ذلك تختبر نـوع الإجـراءات التـي يستخدمها المعالج تلك المهام.

(راجع الفصل "١" لمزيد من التفـاصيل عـن اسـتخدام Database Wizards وAccess Wizards أخرى). وبالمثل، تعد الشفرة أنيقة جداً. تعني كلمة Elegant كما هي مطبقـة علـى البرمجة أن الشفرة لها كفاءة عالية لأنها تستخدم أقل كم من الذاكرة وأقل عـدد مـن العبـارات لتحقيق المهمة. تعد الشفرة الأنيقة مثالية للأداء الممتاز لأنها تستخدم الطرق وأبنية التحكم التـي تحقق المهمة في أقصر وقت.

تنقل النموذج ومجموعة السجلات

عندما تقوم بفتح نموذج منضم، يفتح Access مجموعة سجلات النموذج الموجودة فـي الذاكـرة ويعرض السجلات في النموذج باستخدام محدد سجل النموذج لتشير إلى سجل النموذج الحـالي. فعندما تفتح نموذج منضم، يوجد هناك كائنين تستطيع أن تعالجهما في إجـراء VBA. النمـوذج

ومجموعة سجلات النموذج تستطيع أن تشير إلى مجموعة السجلات مباشرة باستخدام خاصية Recordset Clone للنموذج، التي تعني أنك يمكن أن تستخدم الخصائص والطرق الخاصة بكلاً من كائن Form (راجع الفصل ٥)، وكائن Recordset (راجع الفصل ٦) عندما تكتب الإجراءات التي تعالج السجلات في نموذج.

وعندما ينقص واحد من تلك الكائنات الخاصية أو الطريقة التي تريدها، يمكن أن تستخدم الكائنات الأخرى بدلاً منها. على سبيل المثال، لا يكون لكائن Form خاصية أو طريقة ليحدد عدد السجلات في نموذج منضم، بينما يمكن أن تستخدم خاصية Record Count الخاصة بكائن Recordset. يعد تعلم كيف تستخدم كلاً من كائنات Form وRecordset في الإجراءات مهارة هامة.

ولكي تعمل مع مجموعة سجلات للنموذج في إجراء، قم بتعريف متغير للكائن وقم بالإشارة إلى متغير الكائن لمجموعة سجلات النموذج باستخدام خاصية Recordset Clone الخاصة بالنموذج.

تعد العبارات هي:

```
Dim rst As Recordset, frm As Form
Set frm = Fotrms! Formname
Set rst = frm, RecordsetClone
```

تعد Formname هي اسم النموذج. تقوم عبارة Set الثانية بإنشاء مرجع خاص لكائن Recordset الخاص بالنموذج المسمى recordsetclone وكما تعلمت في الفصل ٦ recordsetclone لها مؤشر السجل الحالي الخاص بها، إذن فهي تستطيع أن تشير إلى سجل مختلف عن السجل الذي تم عرضه في النموذج. وبالتالي يمكن أن تستخدم recordsetclone كي تعمل مع سجل أخر في مجموعة سجل لنموذج بينما يستمر النموذج في عرض سجل خاص. فعندما تنشئ أول مرة recordsetclone، لم يتم تعريف مؤشر السجل الحالي الخاص بها، تستطيع أن تعين مؤشر السجل الحالي في الإجراء.

ملاحظة من أجل التجربة العملية مع التقنيات الموصوفة في هذا الفصل، قم بإنشاء نسخة جديدة لقاعدة بيانات نموذج Northwind المسماة Northwind-Ch11 واعمل خلال الخطوات والأمثلة.

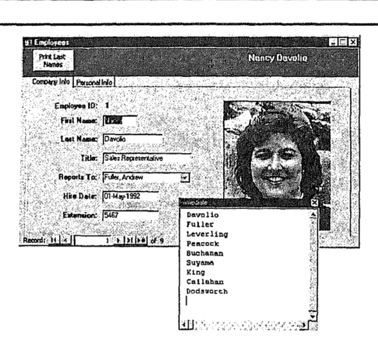
إنشاء إجراء فرعي لتظهر مجموعة السجلات

وكمثال لإنشاء واستخدام recordsetclone. سنقوم بإنشاء إجـــراء لتطبـــع الأســـماء الأخـــيرة للموظفين في جدول الموظفين افتح نموذج Employees في عرض Design. ففي مقطـــع رأس الصفحة، ضع زر أمر يسمى cmdRecordset مع التعليـــق Print Last Names. انقـــر زر خاصية On click وانقر زر Build لتعرض الوحدة النمطية للنموذج. قم بإنشاء إجـــراء حـــدث cmdRecordset-Click الواضحة بالأسفل.

```
Private Sub cmdRecordset_Click()
    Dim rst As Recordset
    Set rst = Me.RecordsetClone
    rst.MoveFirst
    Do
        Debug.Print rst!LastName
        rst.MoveNext
    Loop Until rst.EOF
End Sub
```

يقوم ذلك الإجراء بإنشاء recordsetclone، ويستخدم طريقة MoveFirst كي يعيـــن مؤشـــر السجل الحالي recordsetclone إلى السجل الأول، وبعد ذلك يثبت من خلال مجموعة السجلات يمر كل واحد خلال Loop ويطبع الاسم الأخير في السجل الحالي recordsetclone ويســـتخدم طريقة Move Next لتحرك مؤشر السجل الحالي إلى السجل التالي.

احفظ النموذج وقم بالتبديل إلى عرض Form. افتح إطار Immediate وانقـــر زر الأمـــر الجديد. يطبع إطار Immediate الأسماء الأخيرة لجميع الموظفين كدليل على أن الإجـــراء قـــد ظهر خلال جميع السجلات في مجموعة سجلات النموذج بينما يستمر النموذج في عرض ســـجله الأول "راجع الشكل ١١-١".



الشكل ١١-١ يقوم زر Print Last Names بتشغيل الإجراء الذي يستخدم recordsetclone ويظهر خلال مجموعة سجلات النموذج بينما يعرض النموذج السجل الأول به.

إنشاء إجراء وظيفي لتظهر مجموعة السجلات

في المثال السابق، نستطيع استخدام Me لكي نشير إلى النموذج لأنه قد تم تخزين إجراء الحدث في الوحدة النمطية للنموذج. ففي المثال التالي، سنقوم بإنشاء إجراء وظيفي معاد استخدامه يظهر خلال مجموعة سجلات أي نموذج.

يقوم Form Recordset بإنشاء recordsetclone للنموذج الذي تم تمريره كوسيط للوظيفة وتطبع القيمة الموجودة في الحقل الأول لكل سجل في مجموعة سجلات النموذج.

```
Public Function FormRecordset(frm As Form)
    Dim rst As Recordset
    Set rst = frm.RecordsetClone
    rst.MoveFirst
    Do
        Debug.Print rst(0)
        rst.MoveNext
    Loop Until rst.EOF
End Function
```

ولأننا نريد أن يكون الإجراء الوظيفي يعاد استخدامه، نشير إلى الحقل الأول باستخدام مرجع الفهرس العددي rst Fields (0)، أو ببساطة rst (0) "لأن Fields هي التجميع الافتراضي لمجموعة السجلات". يمكن أن تقوم بتشغيل وظيفة Form Recordset عن طريق تمرير المرجع إلى أي نموذج افتتاحي كوسيط.

قم بإنشاء وحدة نمطية قياسية جديدة مسماة bas Navigation وقم بإدراج إجراء وظيفة Form Recordset الواضح بالأعلى. بعد ذلك، افتح نموذج Customers، واكتب؟ Form Recordset [Forms! Customers] في إطار Immediate، واضغط Enter. لاحظ أنك تقوم بتمرير المرجع إلى النموذج وليس اسم النموذج فعندما تضغط Enter يقوم VBA بطباعة حقل Customer ID إلى إطار Immediate.

ولكي تقوم بتشغيل الوظيفة كإجراء وظيفي للحدث، افتح نموذج Categories في عرض Design واكتب Form Recordset [form]= في حدث On Open. لاحظ أنك تقوم بتمرير Form مثل المرجع للنموذج. (يمكن أن تستخدم أيضاً المرجع Forms! Categories، ولكن لا يمكن أن تستخدم مرجع Me كي تشير إلى النموذج في ورقة الخاصية).

احفظ النموذج وقم بالتبديل إلى عرض Form، يتعرف النموذج على حدث Open ولذا يقوم VBA بتشغيل الإجراء الوظيفي ويطبع حقل Category ID على إطار Immediate.

واجهة التنقل

عندما تعمل تفاعلياً، تستخدم ضغط المفاتيح، وأوامر القائمة، والماوس للتنقل بين عناصر التحكم وبين السجلات، ومن كائن إطار Database إلى كائن آخر. يعد أسهل منهج لجعل التنقل في الواجهة آلياً هو وضع أزرار أمر على النماذج وإنشاء إجراءات يتم تشغيلها عندما تنقر الأزرار.

يوفر لك Access طرق كائن Do cmd للتنقل خلال الواجهة. تقوم تلك الطرق بمضاعفة الإرشادات التفاعلية التي تعطيها عندما تضغط على مفتاح، حرك الماوس وانقر زر الماوس أو اختار أمر قائمة. سنمر خلال خطوات المثال البسيط لكي نقوم بتطوير مجموعة من خطوط الإرشاد لكتابة إجراءات يعاد استخدامها.

تنقل النماذج

يعد المنهج الأسهل لجعل التنقل بين النماذج آلياً هو وضع أزرار أمر على النماذج وإنشاء إجراءات يتم تشغيلها عندما تنقر الأزرار. تعد معظم مهمات تنقل النماذج عامة مثل، فتح نموذج من نموذج آخر، فتح وتزامن نموذج أو تقرير لعرض سجلات خاصة، غلق أو إخفاء نماذج، وأداء مهمات عامة أخرى يحتاج استخدامها العديد من المرات في كل تطبيق. ومن أجل إجراءات الزر المستقل، المحمول، والمعاد استخدامه، سنستخدم إجراءات وظيفية للحدث مخزنة في وحدة نمطية قياسية.

فتح نموذج

افترض أنك تعمل مع نموذج وتريد أن تفتح نموذج أخر، على سبيل المثال، عندما تراجع مستهلك يستخدم نموذج Customers، يمكن أن يريد المستخدم أن يفتح نموذج Order. يعد أسهل منهج لجعل هذه العملية آلية هو وضع زر أمر مسمى cmdOrders على نموذج Customers وإنشاء إجراء حدث لحدث زر Click. استخدم طريقة Open Form الخاصة بكائن DoCmd لتفتح النموذج. تعد طريقة Open Form وسيط مطلوب واحد يتبعه ست وسائط اختيارية.

```
DoCmd.OpenForm formname[,view] [,filtername] [,wherecondition] _
[,datamode] [,windowmode] [,openargs]
```

سنوافق على الافتراضات للوسائط الاختيارية (يمكن أن تبحث عن "Open Form" في تعليمات فورية للمعلومات عن تلك الوسائط) واستخدم وسيط Form name لتحدد اسم النموذج الذي ستقوم بفتحه. يعد وسيط Formname تعبير تسلسلي وهو الاسم الصحيح لأي نموذج في قاعدة البيانات الحالية.

(A First Draft) نعد المسودة الأولى للإجراء هي:

```
Private Sub cmdOrders_Click()
    DoCmd.OpenForm "Orders"
End Sub
```

يعمل إجراء الحدث هذا جيداً– فهو يفتح النموذج المحدد. وتعد المشكلة في هذا الإجراء هي أن طريقة Open Form تستخدم اسم النموذج المحدد كوسيط. يسبب استخدام أسماء محددة للكائنات كوسائط للطرق في إجراء ما مشاكل غير ضرورية إذا أردت إعادة استخدام الإجراء. ولكي تعيد استخدام إجراء يحتوي على أسماء محددة للكائنات تحتاج أن تبحث خلال العبارات لتجد الوسائط الحرفية وتبدل كل واحد باسم جديد. سنقوم بعمل ثلاثة تعديلات للإجراء لنسهل إعادة استخدامها:

- بدل الوسيط الحرفي للطريقة بمتغير.
- حرك تعريف المتغير من الداخل إلى قائمة وسيط الإجراء.
- حرك الإجراء إلى الوحدة النمطية القياسية.

(The First Improvement) يعرف التحسين الأول المتغيرات لتحمل الأسماء المحددة ويستخدم المتغيرات كوسائط للطرق بدلاً من الأسماء المحددة. فعن طريق هذا التحسين يصبح الإجراء:

```
Private Sub cmdOrders_Click()
```

```
    Dim strName As String
    strName = "Orders"
    DoCmd.OpenForm strName
  End Sub
```

يعد إجراء cmdorders_Click سهل للغاية حتى أنه يمكن أن تخفي أهمية التعديل بساطة شديدة. تخيل إجراء له عشرات العبارات التي لها طرق بها وسائط. فعن طريق إنشاء مجموعة من المتغيرات لتحمل جميع الأسماء المحددة وتسند المتغيرات إلى الأسماء المحددة عند بداية الإجراء، يمكن أن تعدل بسهولة الإجراء عن طريق النظر في مكان واحد لتغير الأسماء.

"The Second Improvement": يأخذ التحسين الثاني الأسماء المحددة خارج الإجراء عن طريق إنشاء وسائط للإجراء. فعن طريق هذا التحسين، قم بتمرير الأسماء المحددة كوسائط عندما تستدعي الإجراء. فإذا كان الإجراء إجراء حدث ستحتاج أن تغير نوع الإجراء أيضاً، لأنك لا تستطيع إنشاء وسائطك الخاصة بإجراء حدث. (يعد لإجراء حدث بناء جملة سبق تعريفه، بما في ذلك ذاكرة تسمية، ووسائط معرفة سابقاً لتمرير المعلومات للأمام والخلف بين Access وإجراء الحدث). ولكي تقوم بإنشاء الوسائط الخاصة بك لإجراء تم إطلاقه بواسطة حدث، تحتاج أن تستخدم إجراء وظيفي.

يمكنك أن تغير اسم زر الأمر ووظيفة الحدث لتعكس طبيعته العامة. ففي هذا المثال، سنقوم بتغيير الاسم لكل منهم إلى cmdOpenForm. فعن طريق هذا التعديل، يصبح الإجراء:

```
  Private Function cmdOpenForm(strName As String)
     DoCmd.OpenForm strName
  End Sub
```

فعن طريق المتغير المعرف في قائمة الوسيط الخاصة بالإجراء فلن تحتاج بعد إلى عبارة معرفة منفصلة في الإجراء. لن يظهر على الإطلاق الاسم المحدد للنموذج، لذا يمكن أن تستخدم الإجراء لتفتح أي نموذج. تقوم بإعطاء اسم النموذج عندما تستدعي الإجراء. ففي هذا المثال، لكي تستدعي الإجراء عندما يتم نقر الزر، تقوم بإسناد الوظيفة إلى الحدث عن طريق إدخال =cmdOpenForm("Orders") في ورقة خاصية الزر. فعن طريق تحديد الوسائط في ورقة خاصية الزر، تجنب العمل مباشرة مع الشفرة في الوحدة النمطية. فإذا أردت فتح نموذج أخر، فتحتاج فقط أن تقوم بالتغيير في ورقة خاصية الزر.

"The Third Improvement": يخرج التحسين الثالث الوحدة النمطية للنموذج ويخزنه كوظيفة عامة في وحدة نمطية قياسية، فعن طريق إجراء الوظيفة المخزنة في الوحدة النمطية القياسية، يعد خليط زر الأمر/الإجراء الوظيفي معاد استخدامه كاملاً. فعندما تنسخ الزر، يتم نسخ

الإسناد إلى الإجراء الوظيفي أيضاً. وعن طريق الزر الملصق بالنموذج الأخر، يمكن أن تحـــدد النموذج الذي سيتم فتحه في ورقة خاصية الزر، ولن تحتاج أن تفتح الوحـــدة النمطيـــة لتغــير الإجراء.

وعن طريق وضع إجراء وظيفي للحدث مسند إلى عناصر تحكم في وحدة نمطيـــة قياســـية، تجعل الإجراء المحمول مستقل عن النموذج المحدد في التطبيق، ويعمل لصق عنصر التحكم على لصق الإسناد إلى الإجراء أيضاً، ويستمر عنصر التحكم في العمل طالما تحتوي قاعدة البيانـــات على الوحدة النمطية القياسية. (يمكن أن يتم تخزين الوحدة النمطية القياسية فـــي قـــاعدة بيانـــات أخرى، ويمكن أن يستمر التحكم في العمل إذا قمت بتعيين مرجع لقاعدة البيانات التـــي تحتـــوي على الوحدة النمطية القياسية، راجع الفصل ١٤ للحصول على معلومات عن إعداد المراجع.

ملاحظـــة عندما تخزن إجراء وظيفي للحدث في وحدة نمطية قياســـية، لا يمكـــن أن تستخدم Me لتشير إلى النموذج. بينما عندما يكون النموذج الكائن النشـــيط يمكن أن تشير إلى النموذج باستخدام خاصية Active Form الخاصة بكائن Screen.

يعد خليط زر الأمر/إجراء وظيفي المستقل والمحمول لفتح نموذج هو:

Control وهو زر أمر مسمى cmdOpenForm له خاصية OnClick المعينة عـــن طريق =OpenAForm(Formname) حيث يعد Formname تعبير تسلســـلي يقيـــم اسم النموذج، مثل "Orders".

Event function وظيفة الحدث الآتية المخزنة فـــي الوحـــدة النمطيـــة القياســـية basNavigation:

```
Public Function OpenAForm(strName As String)
    DoCmd.OpenForm strName
End Sub
```

خطوط إرشاد إنشاء إجراءات قابلة لإعادة الاستخدام

يعد الآتي خطـــوط إرشـــاد إنشـــاء إجـــراءات لكـــي تقـــوم بتكبـــير قـــدرة إعـــادة استخدامهم . تعد تلك الاقتراحات خطـــوط إرشـــاد فقـــط، وليســـت قواعـــد صلبـــة يجب أن يتبعها شيء دائماً. علـــى ســـبيل المثـــال، إذا كـــان إجـــراء حـــدث فريـــد بالنسبة لنموذج محدد، قد لا تحتاج أن تذهب إلى ما بعـــد خـــط الإرشـــاد الأول.

- لا يستخدم أسماء محددة أو تعبيرات حرفية كوسائط للطرق، عرف المتغيرات واستخدم متغيرات كوسائط بدلاً من ذلك.
- عرف المتغيرات بطريقة من الطرق:
- في العبارات المعرفة داخل الإجراء وقم بإسنادهم إلى التعبيرات الحرفية.
- كوسائط للإجراء. مرر التعبيرات الحرفية إلى داخل الإجراء. غير إجراء حدث إلى إجراء وظيفي لكي تقوم بتمرير تعبيرات حرفية كوسائط.
- خزن الإجراءات القابلة لإعادة الاستخدام في الوحدات النمطية القياسية بدلاً من تخزينهم في الوحدات النمطية للنموذج أو التقرير عندما تريد أن تجعل الإجراءات مستقلة عن أي نموذج وسهل حملها للنماذج الأخرى. غير إجراء حدث إلى إجراء وظيفي لكي تخزنه في وحدة نمطية قياسية.

إخفاء وإظهار نموذج

عندما تفتح أو تغلق نموذج، يمكن أن يكون هناك مهمات إضافية يحتاج الإجراء إلى تنفيذها. على سبيل المثال، واحد من خطوط إرشاد التطبيق التي يتبعها العديد من المبرمجين لكي يتجنبوا مشاكل تكامل البيانات وهي أن تسمح للمستخدمين بتغيير البيانات في نموذج فردي فقط في وقت ما. وتعد الطريقة المتاحة لتنفيذ الإرشاد هذا هي إخفاء النموذج الأول عندما تفتح النموذج الثاني وبعد ذلك عندما تغلق النموذج الثاني يجب أن يعرف الإجراء الذي يغلق النموذج الثاني أي نموذج سيقوم بإظهاره.

تظهر هنا مشكلة هامة لأنه يمكن بالضبط فتح النموذج الثاني من نماذج عديدة مختلفة معتمداً على كيف تصمم مسارات التنقل. على سبيل المثال، يمكن أن تصمم التنقل حتى يستطيع المستخدم فتح نموذج Orders من Main Switchboard، أو نموذج Customer Orders، أو نموذج Customers. يحتاج النموذج الثاني أن يتذكر أي نموذج قد قام بفتحه حتى يعرف أي نموذج سيقوم بإظهاره عندما تغلقه. تعد خاصية Tag مثالية لهذا الغرض. سنقوم بإنشاء زوج من الإجراءات التي تستخدم خاصية Tag للنموذج الذي تفتحه لتخزن اسم النموذج الذي قام بفتحه. سنقوم بتسمية النموذج الأول "opener form" والنموذج الثاني "opened form".

تم إسناد وظيفة حدث OpenHide إلى زر الأمر على النموذج الأول.

```
Public Function OpenHide (strName As String)
    Dim strHide As String
    strHide = Screen.ActiveForm.Name
    Screen.ActiveForm.Visible = False
    DoCmd.OpenForm strName
```

```
    Screen.ActiveForm.Tag = strHide
  End Function
```

يحمل متغير strHide اسم النموذج الذي قام بالفتح باستخدام كائن Screen لكي يشير إليـــه. فعقب تخزين اسم النموذج الذي يقوم بالفتح في متغير StrHide. يقوم الإجراء بإخفاء النمـــوذج الذي يقوم بالفتح. يقوم الإجراء بفتح النموذج الثاني ويسند خاصية Tag للنموذج الذي يقوم بالفتح إلى اسم النموذج الذي يقوم بالفتح.

عندما تغلق النموذج الثاني عن طريق نقر زر أمر عليه، قم بإظهار النموذج الذي يقوم بالفتح باستخدام إجراء CloseUnhide.

```
  Public Function CloseUnhide()
    Dim strUnhide As String
    If IsNull(Screen.ActiveForm.Tag) Then
      DoCmd.Close
    Else
      strUnhide = Screen.ActiveForm.Tag
      DoCmd.Close
      DoCmd.SelectObject acForm, strUnhide
    End If
  End Function
```

يقوم إجراء CloseUnhide أولاً باختبار خاصية Tag للنموذج الذي قد تم فتحـــه. تحتـــوي خاصية Tag على اسم النموذج الذي قام بالفتح (المختفي) إلا إذا تم فتح النموذج الثاني بطريقـــة أخرى لا تعين خاصية Tag الخاصة بها. ففي هذه الحالة، تعد خاصيـــة Tag هـــي Null. وإذا كانت خاصية Tag هي Null، ستغلق طريقة Close بسهولة النموذج، وإلا يحمل الإجراء اســـم النموذج الذي قام بالفتح في متغير strUnhide. ويغلق النموذج الذي تم فتحه باستخدام طريقـــة Close، وبعد ذلك سيقوم بإظهار النموذج الذي يقوم بالفتح باستخدام طريقة Select Object.

لكي تختبر الإجراءات، اتبع هذه الخطوات:

١- أدرج الإجـــراءات الوظيفيـــة Close Hide وopen Hide فـــي الوحـــدة النمطيـــة basNavegation.

٢- افتح نموذج Products في عرض Design ضع في رأس النموذج زر أمـــر مســـمى cmdOpenHide، مع التعليق Open Categories. قم بإسناد خاصيـــة زر OnClick إلى =OpenHide("Categories") احفظ النموذج وقم بالتبديل إلى عرض Form.

٣- افتح نموذج Categories في عرض Design وأضف زر أمر مسمى cmdCloseUnhide مع التعليق Close Unhide. قم بإسناد خاصية زر OnClick إلى =CloseUnhide() احفظ وأغلق النموذج.

٤- انقر زر Open Categories على نموذج Products. يتم إخفاء النموذج ويفتح نموذج Categories. انقر زر Close Unhide على النموذج يغلق نموذج Categories ويتم إظهار نموذج Product.

ستتعلم المزيد عن استخدام خاصية Tag لتنشئ خصائص مخصصة في الفصل ١٤.

تزامن النموذجين

عندما تستخدم إجراء لتفتح نموذج، تحتاج عادة أن تعمل على تزامن سجلاته إلى السجل المعروض بواسطة النموذج الأول. على سبيل المثال،عندما تفتح نموذج Customers من نموذج Orders، يمكن أن تستخدم وسائط Where Condition أو Filter Name الخاصة بطريقة Open Form لكي تعرض وتحدد مكان سجل المستخدم المماثل للمستخدم المعروض في نموذج Orders. يعد وسيط Filter Name تعبير تسلسلي وهو الاسم الصحيح للاستعلام في قاعدة البيانات الحالية، ووسيط Where Condition تعبير تسلسلي وهو فقرة SQL WHERE صحيحة بدون كلمة WHERE. يمكن أن تستخدم الوسيط لكي تقوم بتقييد السجلات، إذا قمت بتحديد كلا الوسيطين يطبق Access أولاً Filter Name وبعد ذلك يطبق Where Condition على نتيجة الاستعلام.

سنستخدم وسيط Where Condition لنعمل على تزامن النموذج الذي تم فتحه مع النموذج الحالي، كالآتي:

fieldname=Forms!formname!controlname

ففي هذا المثال، يشير Fieldname إلى الحقل في الجدول الموجود أو استعلام النموذج الذي تريد فتحه، وتشير Controlname إلى التحكم على النموذج الذي يحتوي على القيمة التي تريد أن تتطابق معها. على سبيل المثال، لكي تفتح نموذج Customers الذي يعرض سجل متزامن للسجل الموجود في نموذج Orders، استخدم التعبير:

CustomerID=Forms!Orders!CustomerID

يمكن أن تستخدم كائن Screen للإشارة إلى النموذج النشط كما يلي:

CustomerID=Screen.ActiveForm.CustomerID

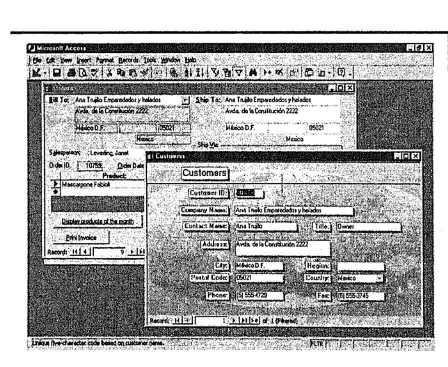
لاحظ أن بناء الجملة الكامل مطلوب على الجانب الأيمن للتعبير، على الرغم من أن Orders يعد النموذج النشيط عندما ينفذ الإجراء Open Form طريقة Open Form. يوضح هذا المثال

الحالة التي يجب أن تستخدم فيها بناء الجملة الكامل لكي تشير إلى التحكم في الكائن النشط. لاحظ أيضا أن بناء الجملة القصير مطلوب على الجانب الأيسر للتعبير. يستخدم Where Condition بناء الجملة المطلوب عن طريق SQL ليعمل على تزامن السجلات.

وعلى سبيل المثال، سنستخدم إجراء حدث يفتح ويعمل على تزامن نموذج Customers لنموذج Orders. افتح نموذج Orders في عرض View وأضف زر أمر مسمى cmdViewCustomer، مع التعليق the form on View Customer. انقر زر Build المجاور لخاصية زر OnClick أدرج إجراء الحدث التالي:

```
Private Sub cmdViewCustomer_Click()
    Dim strForm As String
    Dim strWhere As String
    strForm = "Customers"
    strWhere = "CustomerID = Forms!Orders!CustomerID"
    DoCmd.OpenForm formname:=strForm, wherecondition:=strWhere
End Sub
```

احفظ النموذج وقم بالتبديل إلى عرض Form. استخدم أزرار التنقل الافتراضية لتحدد الترتيب المستخدم. انقر زر View Customer. يتم فتح نموذج Customers، ويحدد ترتيب المستخدمين المختلفين. لا يبقى نموذج Customers متزامناً إلا إذا قمت بنقر زر Customers مرة ثانية.



الشكل ١١-٢ يقوم زر View Customer بتشغيل إجراء يفتح ويتزامن مع نموذج Customers.

الحفاظ على النماذج المتزامنة: عندما تنتقل إلى سجل أخر في نموذج Orders، يستمر نموذج Customers في عرض Customer السابق ولا تعمل على إعادة تزامنه آلياً. سنقوم بإنشاء إجراء حدث لتعيد تزامن نموذج Customers عندما تنتقل إلى سجل مختلف في نموذج Orders. فعندما تنتقل إلى سجل مختلف يتعرف النموذج على الحدث Current. يستخدم إجراء حدث Form_Current الخاص بنموذج Orders الواضح بالأسفل وظيفة Is Loaded المخصصة (مخزنة في الوحدة النمطية Utility Function) لتعيين إذا كان نموذج Customers مفتوح أم لا.

```
Private Sub Form_Current()
    Dim strForm As String
    Dim strWhere As String
    strForm = "Customers"
    strWhere = "CustomerID = Forms!Orders!CustomerID"
    If IsLoaded (strForm) Then
        DoCmd.OpenForm formname:=strForm,
  wherecondition:=strWhere
    End If
 End Sub
```

إذا تم فتح نموذج Customers، يستخدم الإجراء طريقة Open Form إعادة تزامن النموذج وإلا يتم انتهاء الإجراء.

انقر في نموذج Orders وقم بتبديله إلى أسلوب عرض Design قم بنقر زر Build المجاور لنموذج OnCurrent Property وقم بإدراج إجراء Form_Current. قم بحفظ النموذج وبدله إلى طريقة عرض Form. باستخدام أزرار التنقل الافتراضي قم بعمل استعراض إلى ترتيب المستخدم المختلف. تزامن نموذج Customers.

إغلاق النموذج المتصل: عندما تقوم بإغلاق النموذج Orders يجب إغلاق أيضاً النموذج Customers (إذا كان مفتوح). يعد إجراء حدث Form_Close هو حدث Close لنموذج Orders الذي يقوم بغلق نموذج Customers إذا كان مفتوحاً.

```
 Private Sub Form_Close()
    Dim strForm As String
    strForm = "Customers"
    DoCmd.Close acForm, strForm
 End.Sub
```

إنك لست في حاجة إلى تحديد ما إذا كان نموذج Customers مفتوح قبل تشغيل طريقة Close لتقوم بإغلاقه وذلك بسبب الطبيعة الاستثنائية لطريقة Close. حيث أن طريقة Close لا تخطئ إذا قمت بتحديد الاسم لكائن لم يتم فتحه أو لا يوجد.

انقر في نموذج Orders وقم بتبديله إلى طريقة عرض Design انقر زر Build المجاور لنموذج خاصية OnClose وقم بإدراج إجراء Form_Close. احفظ النموذج وبدله إلى طريقة عرض Form. انقر زر Close الافتراضي على النموذج. يتم إغلاق كل من نماذج Orders وcustomers.

التنقل بين عناصر التحكم

عند العمل تفاعلياً سوف تستخدم ضغط المفاتيح، وأوامر القائمة والماوس لتتحرك من خلال عناصر التحكم على النماذج. سوف نقوم بإنشاء إجراءات لجعل التحكم آلياً إلى عناصر تحكم محددة على نماذج ونماذج فرعية وإلى عناصر محددة داخل سجلات من أجل هذا الاستعمال، سوف نضع أزرار الأمر على نموذج Orders ونقوم بإنشاء إجراءات حديثة للتنقل بين عناصر التحكم على نموذج Orders ونموذج Quarterly Orders.

الانتقال إلى تحكم محدد على النموذج النشط

يمكنك أن تنقل التركيز إلى تحكم محدد على النموذج النشط في طريقتين:

- استخدام طريقة Go to Control للكائن DoCmd.
- استخدام طريقة التحكم SetFocus.

استخدام طريقة Go to Control يعد بناء الجملة لطريقة Go to Control هي:

```
DoCmd.GoToControl controlname
```

يعد Controlname هو تعبير تسلسلي وهو اسم عنصر التحكم على النموذج النشيط أو ورقة البيانات. يجب أن تستخدم فقط اسم عنصر التحكم، باستخدام المرجع المؤهل الكامل الذي يسبب خطأ تشغيل.

يحرك إجراء حدث Cmd to Control_Click التركيز إلى عنصر تحكم على النموذج الأساسي الخاص Orders.

```
Private Sub cmdToControl_Click()
    Dim strControl As String
    strControl = "EmployeeID"
    DoCmd.GoToControl strControl
End Sub
```

يتطلب تحريك التركيز إلى عنصر محدد على نموذج فرعي استخدام طريقة Go to Control مرتين. حرك التركيز إلى عنصر تحكم نموذج فرعي بالطريقة الأولى وبعد ذلك إلى عنصر

التحكم على النموذج الفرعي بالطريقة الثانية. يتم تناول هذا التنقل عن طريق إجراء cmdToControlOnSubform_Click procedure.

```
Private Sub cmdToControlOnSubform_Click()
    Dim strControl As String, strSubformControl As String
    strControl = "Discount" : strSubformControl = "Orders Subform"
    DoCmd.GoToControl strSubformControl
    DoCmd.GoToControl strControl
End Sub
```

يأخذ تحريك التركيز إلى عنصر التحكم على النموذج الذي يقوم بالفتح وليس نشيطاً أيضاً خطوتين، كما هو واضح في إجراء cmdToOtherFormControl_Click procedure. تستخدم الخطوة الأولى طريقة Select Object الخاص بكائن DoCmd لتنشيط النموذج الذي يقوم بالفتح وتستخدم الخطوة الثانية طريقة Go to Control لتتحرك إلى عنصر التحكم.

```
Private Sub cmdToOtherFormControl_Click()
    Dim strControl As String, strForm As String
    strForm = "Quarterly Orders" : strControl = "Country"
    DoCmd.SelectObject acForm, strForm
    DoCmd.GoToControl strControl
End Sub
```

افتح نموذج Orders في عرض Design. قم بإنشاء ثلاثة أزرار أوامر على نموذج Orders، وقم بإعطائهم الخصائص التالية الخاصة Name, Caption, OnClick:

اسم الزر	التعليق	إجراء الحدث
CmdToControl	To Control	CmdToControl_Click()
CmdToControlOnSubform	To Control On Subform	CmdToControlOnSubform_Click()
CmdToOtherFormControl	To Control On Other Form	CmdToOtherFormControl_Click()

احفظ النموذج والتبديل إلى عرض Form. افتح نموذج Quarterly Orders. اختبر أزرار الأمر الجديدة على نموذج Orders على سبيل المثال، يتحرك النقر على زر To Control on SubForm إلى عنصر تحكم Discount الخاص بالسجل الأول في النموذج الفرعي "راجع الشكل ١١-٣".

الشكل ١١-٣ انقر زر To Control On SubForm لتشغيل إجراء يحرك التركيز أولاً إلى عنصر تحكم النموذج الفرعي وبعد ذلك إلى عنصر تحكم Discount الخاص بالسجل الأول في النموذج الفرعي.

استخدام طريقة **SetFocus** تعد الطريقة الثانية لتحريك التركيز إلى عنصر تحكم محدد على النموذج النشط الذي يستخدم طريقة عنصر تحكم SetFocus. تعد بناء جملة SetFocus هي:

object.SetFocus
Object is a Form or a Control object.

يحرك كائن Control الخاص بطريقة SetFocus التركيز إلى عنصر التحكم المحدد على النموذج النشيط أو الحقل المحدد على ورقة البيانات النشيطة عندما يكون الكائن هو كائن Form، تعتمد النتيجة على إذا كان النموذج له أي عناصر تحكم تستطيع استقبال التركيز أم لا. فإذا كان للنموذج عناصر تحكم مع مجموعة خاصية Enabled إلى True، تحرك طريقة SetFocus التركيز إلى عنصر التحكم الأخير على النموذج الذي له التركيز، وإلا ستقوم طريقة SetFocus بتحريك التركيز إلى النموذج نفسه. فعندما تنفذ VBA عبارة Object SetFocus تتطلب VBA أن يحرك الكائن التركيز إلى ذاته.

يحرك إجراء حدث CmdSet Control_Click التركيز إلى مربع التحرير والسرد على النموذج الأساسي.

```
Private Sub cmdSetControl_Click()
    Dim cbo As ComboBox
    Set cbo = Forms!Orders!EmployeeID
    cbo.SetFocus
End Sub
```

ولأن SetFocus تعد طريقة خاصة بكائن Control أو كائن Form، تقوم إجراءات الحدث بإنشاء متغيرات كائن لتشير إلى الكائنات. تشير لوحة مفاتيح Set في عبارات التعيين للمتغيرات إلى كائنات محددة. استخدم المرجع المؤهل الكامل لتشير إلى النماذج وعناصر التحكم.

فعن طريق نموذج Orders في عرض Design، ضع ثلاثة أزرار أمر على النموذج، وقم بإعطائهم الخصائص التالية Name وCaption وOnClick

اسم الزر	التعليق	إجراء الحدث
CmdSetControl	SetFocus To Control	CmdSetControl_Click()
CmdSetControlOnSubform	SetFocus To Control On Subform	CmdSetControlOnSubform_Click()
CmdSetOtherFormControl	SetFocus To Control On Other Form	CmdSetOtherFormControl_Click()

احفظ النموذج وقم بالتبديل إلى عرض Form. فعن طريق نموذج Quarterly Orders المفتوح قم باختيار الأزرار الثلاثة الجديدة. على سبيل المثال، يحرك نقر زر SetFocus To Control On Subform التركيز أولاً إلى عنصر تحكم النموذج الفرعي (راجع الشكل ١١-٤).

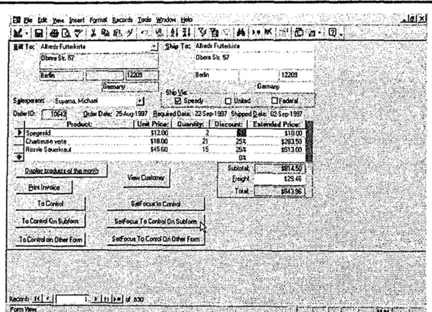
التحرك داخل السجل

عندما تعمل تفاعلياً يمكن أن تستخدم لوحة المفاتيح لتحرك التركيز بين عناصر التحكم الخاصة بالنموذج النشيط. يمكن أن تستخدم عبارة SendKeys لتضاعف ضغط المفاتيح التي تحرك التركيز. يعد بناء الجملة الخاصة بعبارة SendKeys هو:

```
SendKeys string, wait
```

يعد String تعبير تسلسلي يحدد ضغط المفاتيح التي تريد إرساله. يعد وسيط Wait الاختياري True إذا كان يجب معالجة ضغط المفاتيح قبل تنفيذ العبارة التالية وFalse إذا تم تنفيذ العبارة التالية فوراً عقب إرسال المفاتيح. على سبيل المثال، الكلمة التي تنقل التركيز إلى عنصر التحكم الأول في السجل الحالي وينتظر معالجة ضغط المفاتيح هو:

```
SendKeys "{home}", True
```



الشكل ١١-٤

انقر زر SetFocus To Control On SubForm لتشغل إجراء يستخدم طريقة SetFocus لتنقل التركيز أولاً إلى عنصر تحكم النموذج الفرعي وبعد ذلك إلى عنصر تحكم الخاصة Discount بالسجل الأول في النموذج الفرعي.

التنقل بين سجلات نموذج

عندما تعمل تفاعلياً تنتقل بين السجلات طبقاً لمكانهم الطبيعي داخل مجموعة السجلات. تسمى هذه العملية Physical navigation. يعد السجل الذي تنتقله ليصبح سجل حالي Current record هو السجل الذي تعدله عن طريق الماوس التالي أو إجراءات لوحة المفاتيح. يمكن أن تستخدم طريقة Go To Record الخاصة بكائن DoCmd لتضاعف تأثير نقر زر التنقل الافتراضي في أسفل الركن الأيسر للنموذج (أو اختيار أمر Go To على قائمة Edit ويعد ذلك اختيار واحد من الأوامر الفرعية على القائمة الفرعية الخارجية. يعد بناء جملة طريقة Go To Record هو:

DoCmd.GoToRecord [objecttype, objectname][,record][,offset]

حيث:

- يعد Object type واحد من الثوابت الحقيقية:

  acTable, acQuery, acForm, acServerView, or acStoredProcedure

- يعد Object name تعبير تسلسلي اختياري وهو الاسم الصحيح لكائن خاص بالنوع المحدد.

- يعد Record واحد من الثوابت الحقيقية:
  acPrevious, acNext (the default), acFirst, acLast, acGoTo, or acNewRec
- يعد Offset تعبير تعددي يمثل رقم السجلات التي ستنتقل للأمام إذا قمت بتحديد acNext، ارتجع إذا حددت acPrevious أو رقم سجل صحيح إذا حددت acGo To.

تعد جميع الوسائط اختيارية. فإذا قمت بإلغاء وسائط Object name وobject type، يتم افتراض الكائن النشيط. فإذا قمت بإلغاء وسيط Record، يعد الثابت الافتراضي هو acNext.

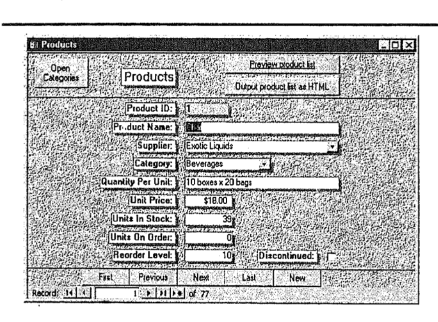
إنشاء أزرار تنقل مخصصة

سنقوم بإنشاء مجموعة من أزرار التنقل المخصصة للنموذج، ولكي تجعل الزرار قابلة إعادة الاستخدام على النماذج الأخرى، نجعل الأزرار آلية عن طريق الإجراءات الوظيفية للأحداث الواضحة بالأسفل، وهي مخزنة في الوحدة النمطية القياسية الجديدة bas Navigation Buttons.

```
Public Function FirstRecord()
    DoCmd.GoToRecord Record:= acFirst
End Function
Public Function PreviousRecord()
    DoCmd.GoToRecord Record:= acPrevious
End Function
Public Function NextRec()
    DoCmd.GoToRecord Record:= acNext
End Function
Public Function LastRecord()
    DoCmd.GoToRecord Record:= acLast
End Function
Public Function NewRec()
    DoCmd.GoToRecord Record:= acNewRec
End Function
```

ملاحظة يتم اختصار الإجراءات الوظيفية للأزرار Next وnew لأن Next وRecord وnew Record يعدان أسماء للخصائص وهم لا يعدان أسماء صحيحة للإجراءات.

قم بإنشاء مجموعة من أزرار الأمر الخمسة في مقطع تذييل الصفحة الخاص بنموذج Products المسماة cmdFirst وcmdPrevious وcmdNext وcmdLast وcmdNew "وعين طريق تعليقات مناسبة" قم بتعيين الإجراءات الوظيفية للأحداث الواضحة بالأعلى. تأكد من أنك تغيير خاصية Visible التي تخص Form Footer إلى Yes. احفظ النموذج وقم بتبديله إلى عرض Form "راجع الشكل ١١-٥" اختبر الأزرار الجديدة.



الشكل ١١-٥
أزرار التنقل المخصصة.

عندما تختبر الأزرار تجد أن نقر الزر First وبعد ذلك نقر الزر Previous يجعلان إجراء Previous Record يفشل "راجع الشكل ١١-٦ أ". يعد سبب الفشل هو أنه بعد نقر زر First يعد السجل الحالي هو السجل الأول في مجموعة السجلات، لهذا عندما تنقر زر Previous فأنك تحاول أن تنتقل خلف حدود مجموعة السجلات.

يفتح نقر زر Last وبعد ذلك نقر زر Next نموذج فارغ. فعقب نقر زر Last وبعد ذلك نقر زر Next، يعد السجل الحالي هو السجل الجديد الذي يلي السجل الأخير في مجموعة السجلات.

التعامل مع أخطاء وقت التشغيل

يوجد هناك طريقتين للتعامل مع أخطاء وقت التشغيل التي تظهر عندما تحاول أن تنتقل خلف الحدود العلوية لمجموعة السجلات:

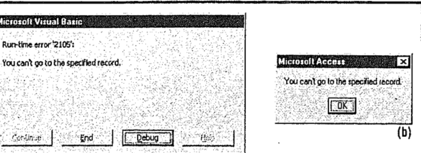
- يمكن أن تعمل على تضمين شفرة معالجة الخطأ لتتجنب أخطاء وقت التشغيل.
- يمكن أن تنشئ مجموعة أزرار تنقل "Smart" التي تعطل أي زر عندما تنقره مما يسبب خطأ وقت التشغيل.

إضافة شفرة معالجة الخطأ

وكمثال لاستخدام معالجة الخطأ لتجنب خطأ وقت التشغيل، قمنا بإضافة معالجة خطأ مخصص إلى الإجراء الوظيفي Previous Record لإنهاء الإجراء بدون فشل عندما تظهر الأخطاء. يعد الإجراء المعدل Previous Record الواضح بالأسفل معالج متمكن للخطأ يعرض رسالة خطأ افتراضية عندما تحاول أن تنتقل خلف السجل الأول.

```
Public Function PreviousRecord()
    On Error GoTo PreviousRecord_Err
    DoCmd.GoToRecord Record:=acPrevious
PreviousRecord_Exit:
Exit Function
PreviousRecord_Err:
    MsgBox Err.Description
    Resume PreviousRecord_Exit
End Function
```

عقب تعديل إجراء Previous Record، لا يزال نقر First وبعد ذلك نقر زر Previous يحدث خطأ وقت التشغيل كما سبق. بينما، في هذه المرة يتم تعيين تصيد الخطأ، ينتقل عنصر التحكم إلى شفرة معالجة الخطأ. يقوم معالج الخطأ بعرض الرسالة الواضحة في الشكل "١١-٦-ب" ويخرج الإجراء بدون محاولة تنفيذ طريقة Go TO Record. وكبديل لذلك، يمكن أن تلغى عبارة MsgBox وبسهولة لا يوجد أي استجابة عندما تصل إلى السجل الأول فانقر زر Previous. يمكن أن تكتب شفرة معالجة أخطاء مماثلة لأزرار التنقل الأخرى.



الشكل ١١-٦

عندما تريد أن تتنقل خلف حدود مجموعة السجلات، تعد معالجة الخطأ الافتراضي هي فشل الإجراء "أ"، ولكن يمكن أن تستخدم معالجة خطأ مخصص "ب" لتتجنب الفشل.

إنشاء أزرار تنقل ذكية

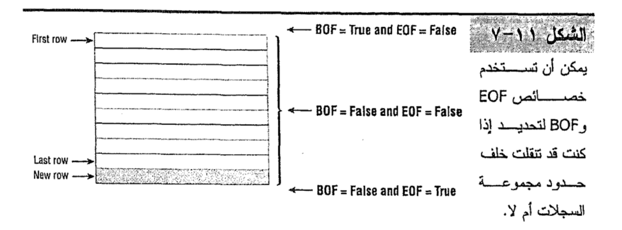
قم بإنشاء إجراء لقوم بتعطيل زر تنقل عندما تنقره مما سيسبب خطأ وقت التشغيل. وبالتالي، إذا كان السجل الأول هو السجل الحالي، يقوم الإجراء بتعطيل زر Previous. سنقوم بتصميم الإجراء لنعطل زر First أيضاً عندما يكون السجل الأول هو السجل الحالي، على الرغم من أن إجراء First Record لا يفشل عندما يتم نقر زر First باستمرار. يخدم زر First المعطل كمساعد مرئي ويعد السجل الأول هو السجل الحالي، يقوم الإجراء بتعطيل أزرار New وLast وFirst.

ولكي تعطل الأزرار المناسبة، يجب أن تعرف إذا كان السجل الحالي هو الأول، الأخير أو السجل الجديد أم لا. يعد تحديد إذا كان السجل الحالي هو سجل جديد أمراً سهلاً لأن النموذج له خاصية New Record التي تستطيع أن تختبرها. بينما، لا توجد خاصية First Record أو Last Record للنموذج، لذا سنحتاج أن نعمل جيداً لنجد إذا كان السجل الحالي هو السجل الأول أو السجل الأخير.

على الرغم من أن ليس للنموذج أي خصائص تستطيع استخدامها لتحديد إذا كان السجل الحالي هو السجل الأول أو الأخير، تقوم مجموعة السجلات الخاصة بالنموذج بذلك.

يوجد بمجموعة السجلات خصائص EOF وBOF التي يمكن أن تستخدمها لتحدد إذا كنت قد انتقلت خلف الحدود أم لا. فإذا قمت بنقل موضع السجل الحالي إلى مكان ما قبل السجل الأول، تعد BOF هي True وإذا كنت قد قمت بنقل موضع السجل الحالي إلى مكان ما بعد السجل الأخير، تعد EOF هي True. "راجع الشكل ١١-٧".

الشكل ١١-٧
يمكن أن تستخدم خصائص EOF وBOF لتحديد إذا كنت قد تنقلت خلف حدود مجموعة السجلات أم لا.



إذا بدأنا بالتنقل في مجموعة سجلات النموذج، سنقوم بإزعاج عرض الشاشة، لذلك بدلاً من هذا سنستخدم خاصية RecordsetClone لإنشاء مؤشر سجل حالي منفصل، اعمل على تزامن RecordsetClone حتى يشير إلى السجل الحالي الذي يعرضه النموذج، واستخدم مؤشر RecordsetClone لتتنقل في مجموعة السجلات. على سبيل المثال، إذا استخدمت طريقة Move Previous التي تخص RecordsetClone لتنقله إلى السجل السابق في مجموعة السجلات ولتجد أن BOF هي True، ففي هذه الحالة يجب أن يعرض النموذج السجل الأول.

يقوم الإجراء الوظيفي Disable Enable بتعطيل وتشغيل أزرار التنقل.

```
Public Function DisableEnable(frm As Form)
'To call the function set the form's OnCurrent
'property to =DisableEnable(Form)
    Dim rstClone As Recordset
'Create a clone of the form's recordset to
'move around in without affecting the form's
'recordset
    Set rstClone = frm.RecordsetClone
'Determine if the current record is the
'new record and if it is, disable the Next
'and New buttons and then exit.
    If frm.NewRecord Then
        frm!cmdFirst.Enabled = True
        frm!cmdNext.Enabled = False
        frm!cmdPrevious.Enabled = True
        frm!cmdLast.Enabled = True
        frm!cmdNew.Enabled = False
```

```
        Exit Function
    End If
'If the current record is not the new record
'enable the New button
    frm!cmdNew.Enabled = True
'If there are no records, disable all
'other buttons
    If rstClone.RecordCount = 0 Then
        frm!cmdFirst.Enabled = False
        frm!cmdNext.Enabled = False
        frm!cmdPrevious.Enabled = False
        frm!cmdLast.Enabled = False
    Else
'Synchronize the current record in the clone
'to be the same as the current record displayed
'in the form.
        rstClone.Bookmark = frm.Bookmark
'Move to the previous record in the clone,
'if the clone's BOF is True, the form must be
'at the first record so disable the First and
'Previous buttons. Otherwise, the form is not
'at the first record so enable the First and
'Previous buttons.
        rstClone.MovePrevious
        If rstClone.BOF Then
            frm!cmdFirst.Enabled = False
            frm!cmdPrevious.Enabled = False
        Else
            frm!cmdFirst.Enabled = True
            frm!cmdPrevious.Enabled = True
        End If
'Resynchronize the current record in the clone
'to be the same as the current record displayed
'in the form.
```

```
        rstClone.Bookmark = frm.Bookmark
    'Move to the next record in the clone,
    'if the clone's EOF is True, the form must be
    'at the last record so disable the Next and
    'Last buttons. Otherwise, the form is not
    'at the first record so enable the Next and
    'Last buttons.
        rstClone.MoveNext
        If rstClone.EOF Then
            frm!cmdNext.Enabled = False
            frm!cmdLast.Enabled = False
        Else
            frm!cmdNext.Enabled = True
            frm!cmdLast.Enabled = True
        End If
      End If
    End Function
```

يتم تشغيل الإجراء عندما يتم فتح النموذج أولاً وعندما تنتقل إلى سجل مختلف "ويتعرف النموذج على حدث Current". ولكي تجعل الإجراء قابل لإعادة الاستخدام على النماذج الأخرى سنقوم بتمرير النموذج إلى الإجراء كوسيط.

تبدأ وظيفة Disable Function بإنشاء RecordsetClone عن طريق مؤشر السجل الحالي المنفصل الخاص بها، يمكن أن يتبدل الإجراء الآن بين النموذج ومجموعة السجلات باستخدام أي خاصية نموذج أو مجموعة سجلات أو أي طريقة تحتاجها. يتحدد الإجراء إذا كان السجل الحالي هو السجل الجديد عن طريق اختبار خاصية New Record الخاصة بالنموذج. وإذا كانت هي السجل الجديد تصبح أزرار Next وNew معطلة وينتهي الإجراء. وإذا لم يكن السجل الحالي هو السجل الجديد، يتحدد الإجراء إذا احتوت مجموعة السجلات على أي سجلات.

يعد تحديد وجود سجلات أمراً ضرورياً إذا حاولنا أن ننتقل في مجموعة السجلات التي ليس بها سجلات على الإطلاق. نسبب بذلك أخطاء وقت التشغيل. يستخدم الإجراء خاصية Record Count التي تخص RecordsetClone لتحديد إذا كان هناك سجلات وتعطيل أزرار First وPrevious وNext وLast إذا لم يكن هناك أي سجلات. فإذا كان هناك سجلات، تعد الخطوة الأولى هي تحديد موقعنا في مجموعة السجلات.

ملاحظـــة ترجع خاصية RecordCount العدد الكلي للسجلات إذا كـانت مجموعـة السجلات هي مجموعة سجلات مكتوبة في جدول. وإذا كـانت مجموعـة السجلات هي مجموعة سجلات كتابة "المجموعـة الديناميكيـة". فـترجع خاصية RecordCount عدد السجلات التي تم الوصول إليـها. وبمجـرد الوصول إلى السجل الأخير في مجموعـة سـجلات كتابـة "المجموعـة الديناميكية"، ترجع خاصية Record Count عدد السجلات الكلـي. تعـد الطريقة الوحيدة لضمان الوصول إلى السجلات هو استخدام طريقة Move-Last الخاصة بكائن Recordset قبل قراءة خاصيـة Record Count. بينما، إذا كان كل ما تريد أن تعرفه هو وجـود أي سـجلات، يمكـن أن تستخدم خاصية RecordCount بدون أن تسـتخدم طريقـة MoveLast. وإذا لم يتم إيجاد أي سجلات، يتم إرجاع خاصيـة RecordCount إلـى صفر، وإلا يتم إرجاع الخاصية إلى عدد صحيح أكبر من صفر.

سنستخدم مؤشر السجل الحالي الذي يخص Recordsetclone لنتنقل في مجموعة السـجلات ونختبر إذا كنا قد نقلنا مؤشر Recordsetclone خلف حدود مجموعة السـجلات أم لا. فنحـن نحتاج طريقة لنعمل على تزامن Recordsetclone للنمـوذج حتـى نسـتطيع أن نبـدأ بـ Recordsetclone وإشارة النموذج للنموذج لنفس السجل. نستخدم خاصية Bookmark. لنقـوم بالتزامن. فعندما تفتح نموذج منضم، تقوم Access آلياً بإسناد إشارة مرجعية فريدة إلى كل سجل في مجموعة السجلات. ولكلاً من النموذج وrecordsetclone خاصيـة Bookmark. ترجـع خاصية Bookmark للنموذج قيمة الإشارة المرجعية للسجل المعروض بواسطة النموذج. وترجع خاصية Bookmark التي تخص Recordsetclone قيمة الإشارة المرجعيـة للسـجل الحـالي الخاص به، حتى يمكن أن تشير إلى Recordsetclone بالسجل المعروض في النموذج باستخدام عبارة التعيين هذه:

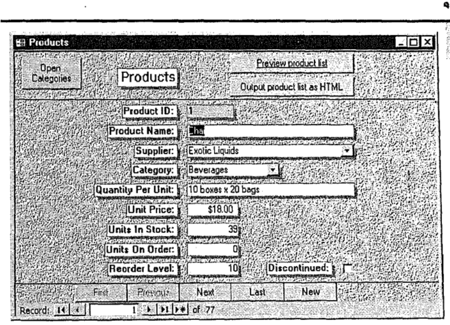
```
rstClone.Bookmark = frm.Bookmark
```

يستخدم إجراء Disable Enable خصائص EOF وBOF لتحديد إذا كان السجل الحالي هـو السجل الأول أم السجل الأخير. ويحدد الإجراء إذا كان السجل الحالي هو السجل الأول باستخدام طريقة Move Previous التي تخص Recordsetclone لنقل مؤشر السجل الحـالي المـاضي Recordsetclone إلى السجل السابق وبعد ذلك اختبار خاصيـة Recordsetclone's BOF. فإذا كانت خاصية Recordsetclone's BOF هـي True، إذاً فلقـد قـامت طريقـة Move Previous بنقلنا إلى موقع السجل الحالي قبل السجل الأول، ويجب أن نكون عند السـجل الأول

قبل النقل. ففي هذه الحالة نقوم بتعطيل الأزرار First وprevious. فإذا كانت خاصية BOF التي تخص Recordsetclone هي False، فنحن لسنا بالسجل الأول قبل النقل، لذا نعمل على تمكين الأزرار First وprevious. يتزامن الإجراء مع Recordsetclone والنموذج ليشير إلى نفس السجل.

تعد الخطوة الأخيرة هي تحديد إذا كان السجل الحالي هو السجل الأخير باستخدام طريقة Move Next التي تخص Recordsetclone لنقل مؤشر السجل الحالي الذي يخص Recordsetclone إلى السجل القادم وبعد ذلك اختبار خاصية Recordsetclone's EOF. فإذا كانت خاصية Recordsetclone's EOF هي True، إذاً تنقلنا طريقة Move Next إلى موقع السجل الحالي بعد السجل الأخير، ويجب أن نكون بالسجل الأخير قبل النقل. ففي هذه الحالة، نعطل الأزرار Last وNext. فإذا كانت خاصية Recordsetclone's EOF هي False، فنحن لسنا بالسجل الأخير قبل النقل، لذا نمكن الأزرار Last وNext فبعد إنهاء الاختبارات ينتهي الإجراء.

قم بإدراج الإجراء الوظيفي Disable Enable في الوحدة النمطية Navigation bas Buttons. وبعد ذلك اعرض نموذج Products في عرض Design. انقر في خاصية On Current الخاصة بالنموذج وقم بإسنادها إلى =DisableEnable(Form). احفظ النموذج وقم بالتبديل لعرض View. اختبر الأزرار على سبيل المثال، عندما يكون السجل الأول هو السجل الحالي للنموذج، يتم تعطيل أزرار First وPrevious (راجع الشكل ١١-٨).

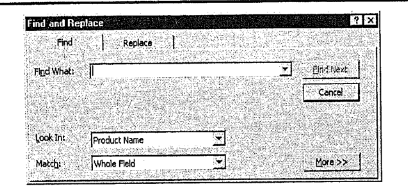


الشكل ١١-٨
يمكن ويعطل إجراء Disable Enable لأزرار الأمر المخصص لتجنب أخطاء وقت التشغيل.

ملاحظة ينشئ استخدام خاصية RecordsetClone الخاصة بالنموذج مرجع جديد لكائن النموذج الموجود Recordset. فعندما تقوم بإنهاء استخدام RecordsetClone. لا تستخدم طريقة Close لتحاول أن تغلق كائن Recordset. لا يسمح لك VBA بإغلاق مجموعة سجلات النموذج بدون إغلاق النموذج وتجاهلها ببساطة أي عبارة تحاول إغلاق RecordsetClone. يمكن أن تسند متغير الكائن إلى Nothing لیقطع الصلة بين المتغيرات وrecordsetClone. ولكن طبيعياً لا توجد أي نقطة في عمل ذلك.

إيجاد سجل محدد

عندما تبحث عن السجلات تفاعلياً، استخدم حوار Edit ⇦ Find" راجع الشكل ١١-٩". ولكي تجعل عملية البحث أسرع، قم بتقييد البحث عن قيم في عنصر تحكم عن طريق تحديد عنصر التحكم قبل عرض حوار Find.



الشكل ١١-٩ أثناء العمل تفاعلياً استخدم حوار Find لتجد سجل له قيمة محددة في الحقل.

فعندما تجعل عملية البحث آلية باستخدام برمجة VBA، يمكن أن تتجنب عرض حوار Find وتجعل تطبيقك أسهل في الاستخدام عن طريق إضافة مربع تحرير وسرد غير منضم لمقطع رأس وتذييل النموذج والسماح للمستخدم ببدء البحث لإنشاء إجراء حدث لتجد سجل محدد مماثل للقيمة المحددة.

يتم تشغيل إجراء الحدث عندما يغير المستخدم القيمة في مربع التحرير والسرد ويتعرف مربع التحرير والسرد على حدث After Up Date. سنراجع ثلاث طرق لنجد سجل محدد:

- ♦ استخدام طريقة Find Record.
- ♦ استخدام طريقة Apply Filter لكائن DoCmd.

♦ استخدام خاصية RecordsetClone.

ولكي تستكشف تقنيات البحث، افتح نموذج Employees في عرض Design وضع مربع تحرير وسرد غير منضم في رأس المقطع. ثم قم بتعيين خصائص التحرير والسرد الآتي:

الاسم	CboFind
نوع مصدر الصف	Table/ Query
مصدر الصف	Employees
عدد العمود	2
عرض العمود	0";0.75"
ضم العمود	1

استخدام طريقة Find Record لكائن DoCmd

يعد أسهل منهج لإيجاد سجل محدد هو إنشاء إجراء حدث لمربع التحرير والسرد الذي يعكس كل خطوة تفاعلية للعملية الخاصة بعبارة VBA. يستخدم إجراء الحدث الواضح بالأسفل هذا المنهج.

```
Private Sub cboFind_AfterUpdate()
    Application.Echo False
    EmployeeID.Enabled = True
    EmployeeID.SetFocus
    DoCmd.FindRecord cboFind
    cboFind.SetFocus
    EmployeeID.Enabled = False
    Application.Echo True
End Sub
```

يبدأ هذا الإجراء عن طريق إغلاق رسم الشاشة أثناء تشغيل الإجراء. ففي انعكاس كل خطوة تفاعلية، يجب أن ينقل الإجراء التركيز إلى عنصر تحكم Employee ID، بينما يتم تعطيل عنصر التحكم هذا، لذا يجب أن يمكن الإجراء أولاً عنصر التحكم. فبعد تمكين عنصر التحكم Employee ID ينقل الإجراء التركيز إليه ويستخدم طريقة Find Record ليجد القيمة الموجودة في مربع التحرير والسرد. وبعد إيجاد القيمة، ينقل الإجراء التركيز للخلف إلى مربع التحرير والسرد، ويعطل عنصر تحكم Employee ID ويفتح رسوم الشاشة للخلف وينتهي.

ملاحظـــة افتراضاً، يتم فتح رسومات الشاشة، ويأخذ Access الوقت لتحديث الشاشـــة لكل عبارة. ولا تأخذ فقط إعادة الرسومات الوقت ولكن أيضاً تجعل الشاشة تضطرب كلما تقوم الشاشة بالتحديث بعد كل عبارة. وعندما تغلق رسومات الشاشة في إجراء VBA يجب أيضاً أن تعود وتفتحها قبل إنهاء الإجراء.

استخدام طريقة Apply Filter لكائن DoCmd

يستخدم منهج ذو كفاءة عالية التصفية لتحديد السجل مباشرة من مجموعة سجل النموذج. تســـمح لك طريقة Apply Filter بتطبيق التصفية على الجدول، أو النموذج أو التقرير لتقييـــد أو تنويـــع السجلات في الجدول أو في مجموعة سجلات النموذج أو القرير الهامة، يمكن أن تحدد اســـتعلام محفوظ كتصفية باستخدام وسيط Filtername. أو يمكن أن تدخل فقرة SQL WHERE "بـــدون كلمة WHERE" في الوسيط Where Condition. ولطريقة Apply Filter ثلاث وسائط:

- يعد FilterName تعبير تسلسلي وهو اسم استعلام أو تصفية تم حفظها كاستعلام يقيد أو ينوع السجلات.
- يعد Where Condition تعبير يقيد السجلات الموجودة في نموذج فقرة SQL WHERE الصحيحة بدون كلمة WHERE.
- يحدد Filtername هل تبحث عن بيانات منسقة (Normal) أو بيانـــات غـــير منســـقة (Server).

يجب أن تحدد على الأقل واحد من بين أثنين من الوسائط فإذا حددت كلا الوسيطين، يطبـــق Access أولاً الاستعلام وبعد ذلك يطبق Where Condition على نتيجة الاستعلام. يعد الطـــول الأقصى الخاص بوسيط Where Condition هـــو ٣٢٧٦٨ حـــرف. "يعـــد وســـيط Where Condition الخاص بإجراء ماكرو Apply Filter المماثل هو ٢٥٦ حرف".

يعد وسيط Where Condition الخاص بتزامن النموذج للقيمة الموجودة في مربع التحريـــر والسرد هو كالأتي:

```
fieldname=Forms!formname!controlname
```

ففي هذا التعبير، يشير Fieldname إلى الحقل الموجود في الجـــدول أو الاســـتعلام الـــهام للنموذج، ويشير Controlname إلى عنصر التحكم على النموذج الذي يحتوي على القيمة التـــي تريد أن تطبقها. على سبيل المثال، لكي تعمل على تزامن نموذج Employees للقيمة المعروضة في مربع التحرير والسرد cobFind، استخدم التعبير:

```
[EmployeeID]=Forms![Employees]![cboFind]
```

يمكن أن تستخدم أيضاً خاصية ME لتشير إلى النموذج كالأتي:

```
[EmployeeID]=Me!cboFind
```

يستخدم إجراء الحدث الموضح بالأسفل هذا المنهج.

```
Private Sub cboFind_AfterUpdate()
    Dim strSQL As String
    strSQL = "EmployeeID = " & Me!cboFind
    DoCmd.ApplyFilter wherecondition:= strSQL
End Sub
```

استخدام Recordset Clone

يستخدم المنهج الأكثر كفاءة recordsetClone الخاص بالنموذج ليشير إلى مجموعة سجلات النموذج. ولكائن recordset عدة طرق يمكن أن تستخدمها لتجد سجل محدد. ولكي تحصل على الوصول إلى تلك الطرق، يمكن أن تستخدم خاصية RecordsetClone للنموذج لتشير إلى مجموعة سجلات النموذج. تعرف العبارات الآتية rst ككائن متغير وتسنده إلى كائن Recordset الخاص بالنموذج بمؤشر السجل الحالي الخاص به:

```
Dim rst As Recordset
Set rst = Me.RecordsetClone
```

يمكن أن نستخدم طريقة Find First الخاصة بكائن Recordset لتنقل مؤشر السجل الحالي الخاص recordsetClone إلى السجل الأول الذي يرضي معيار محدد. يعد بناء جملة طريقة Find First هي:

```
recordset.FindFirst criteria
```

يعد Recordset مرجع لكائن مجموعة سجلات كتابة recordset أو مجموعة داين الموجودة. يعد Criteria تعبير تسلسلي يقيد السجلات في مجموعة السجلات. ويعد وسيط Criteria فقرة SQL WHERE صحيحة بدون كلمة WHERE.

فبعد تشغيل طريقة Find First، يشير recordsetClone إلى إيجاد سجل بينما لم يتغير السجل الحالي المعروض في النموذج. تعد الخطوة الأخيرة هي نقل مؤشر السجل الحالي للنموذج إلى نفس السجل الذي يشير إليه recordsetClone عن طريق إعداد خاصية Bookmark الخاصة بالنموذج لخاصية Bookmark التي تخص recordsetClone:

```
Me.Bookmark = rst.Bookmark
```

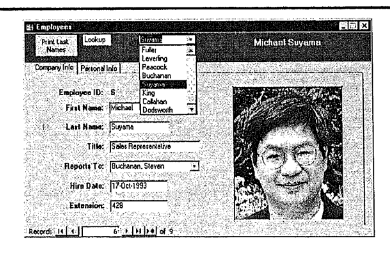
يستخدم إجراء الحدث الواضح بالأسفل هذا المنهج.

```
Private Sub cboFind_AfterUpdate()
    Dim strCriteria As String
    Dim rst As Recordset
    Set rst = Me.RecordsetClone
    strCriteria = "EmployeeID = " & Me!cboFind
    rst.FindFirst strCriteria
    Me.Bookmark = rst.Bookmark
End Sub
```

إذا كان الغرض الوحيد لإجراء الحدث هو إيجاد السجل وعدم أخذ أي إجراءات أخرى باستخدام المتغيرات، فلا تحتاج أي متغيرات على الإطلاق ويمكن أن تقوم بتبسيط الإجراء. افتح نموذج Employee في عرض Design وقم بإدراج الإجراء المبسط cobFind_AfterUpdate الواضح بالأسفل:

```
Private Sub cboFind_AfterUpdate()
    Me.RecordsetClone.FindFirst "EmployeeID = " & Me!cboFind
    Me.Bookmark = Me.RecordsetClone.Bookmark
End Sub
```

احفظ النموذج وبدله إلى عرض Form. حدد موظف في مربع التحرير والسرد للبحث "راجع الشكل ١١-١٠". يعرض الإجراء السجل الخاص بالموظف المحدد.



الشكل ١١-١٠
تستخدم تقنية البحث ذات أعلى كفاءة طريقة Find First التي تخص RecordsetClon.

تراجع البحث

بعد إيجاد سجل معين، يمكن أن تريد أن تتراجع عن البحث وترجع إلى السجل المعروض السابق. ولكي تتراجع عن البحث، تحتاج أن تعرف أي سجل تم عرضه أخر العرض. يمكن أن

تبتعد عن السجل السابق عن طريق حمل قيمة المفتاح الأول الخاص به في متغير لـــه مســتوى وحدة نمطيه. استخدم متغير له مستوى الوحدة النمطية حتى يتوفر المفتاح الأول للسجل لإجــراء التراجع عن البحث. قم بتعديل الإجراء الذي يجد سجل محدد عن طريق تضمين عبارة لتخــزن قيمة المفتاح الأول للسجل الحالي قبل تشغيل العبارات التي تجد السجل المحدد.

ولكي تتراجع عن البحث عندما يستخدم البحث تقنية Recordsetclone، أدخل أولاً عبـــارة التعريف التالية في مقطع Declaration للوحدة النمطية للنموذج:

```
Private LastFind
```

بعد ذلك، قم بتعديل إجراء الحدث cobFind_AfterUpdate لتسند Last Find إلى المفتـاح الأول للسجل الحالي قبل أي عبارة أخرى:

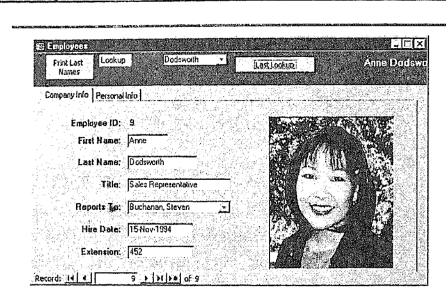
```
Private Sub cboFind_AfterUpdate()
   LastFind = EmployeeID
   Me.RecordsetClone.FindFirst "EmployeeID = " & Me!cboFind
   Me.Bookmark = Me.RecordsetClone.Bookmark
End Sub
```

ففي رأس نموذج Employees ضع زر أمر مسمى cmdUndoFind مـــع التعليـــق Last Lookup وقم بإنشاء إجراء الحدث OnClick الواضح بالأسفل:

```
Private Sub cmdUndoFind_Click()
   Me.RecordsetClone.FindFirst "EmployeeID = " & LastFind
   Me.Bookmark = Me.RecordsetClone.Bookmark
   Me!cboFind = LastFind
End Sub
```

يستخدم إجراء الحدث CmdUndoFind-Click نفس تقنية البحث لإيجاد السجل الذي يطـــابق القيمة المخزنة في متغير Last Find وبعد ذلك أعمل على تزامن مربع التحرير والسرد للســجل المعروض.

احفظ النموذج وبدله إلى عرض Form. حدد موظف في مربع التحرير والسرد للبحث. انقــر زر Last Lookup. يتم عرض الموظف السابق (راجع الشكل ١١-١١).



الشكل ١١-١١ يقوم زر Last Lookup بتشغيل إجراء يستخدم قيمة Employee ID المخزنة في متغير له مستوى وحدة نمطية لتحدد مكان السجل الذي سبق إيجاده.

العمل مع البيانات في الجدول

لقد عملنا في المقاطع السابقة مع السجلات في نموذج يقوم بالفتح. ولقد استخدمنا خاصية RecordsetClone الخاص بالنموذج لنشير إلى كائن Recoedset الخاص بالنموذج حتى نستطيع الوصول إلى الخصائص والطرق لكائن Recordset، وتوجد طريقة أخرى للعمل مع السجلات في كائن Recordset وهي فتح مجموعة سجلات في الذاكرة مباشرة دون العمل مع النموذج على الإطلاق. ويعد الفتح والعمل مع مجاميع السجلات الموجودة في الذاكرة ميزة أداء مميزة لأن Access لا يحتاج أن يأخذ وقت لإنشاء تمثيل مرئي للنموذج على الشاشة.

إنشاء متغيرات مجموعة السجلات

تعد مجموعة السجلات مجموعة من السجلات لجدول أو مجموعة من السجلات الناتجة عن تشغيل استعلام أو عرض أو عبارة SQL التي ترجع السجلات. وعندما تعمل مع بيانات في إجراءات VBA، فإنك تعمل مع مجاميع سجلات. استخدم طريقة Open Recordset لكائن Database لتقوم بإنشاء كائن Recordset جديد مستند على الجدول أو الاستعلام أو العرض أو عبارة SQL. استخدم العبارات الآتية لتنشئ مجموعة سجلات جديدة:

```
Dim rst As Recordset
Set rst = database.OpenRecordset(source, type, options, lockedits)
```

تعد Database مرجع لكائن Database الموجود، ويعد Source تعبير تسلسلي يحدد اسم جدول أو عرض أو استعلام أو عبارة SQL التي ترجع السجلات. استخدم وسائط Type, Options, and Lockedits الاختيارية لتحديد ميزات مجموعة السجلات. راجع الفصل ٦

للحصول على المزيد من المعلومات عن بناء جملة طريقة Open Recordset، والخصائص والطرق الخاصة بأربع أنواع لمجاميع السجلات.

تقوم مجموعة السجلات بإنشاء حدث يوجد في الذاكرة فقط أثناء تشغيل الإجراء. وعندما يتم إنهاء الإجراء، يتوقف متغير كائن Recordset عن التواجد ويتم تدمير كائن Recordset.

فتح مجموعة سجلات لها نوع الجدول

قم بإنشاء وحدة نمطية قياسية جديدة مسماة bas Recordset للأمثلة الموجودة في المقطع. يعد مثالنا الأول هو إجراء Table Recordset الواضح بالاسفل:

```
Public Sub TableRecordset()
   Dim rst As Recordset
   Set rst = CurrentDB.OpenRecordset("Categories", dbOpenTable)
   Do Until rst.EOF
      Debug.Print rst(0), rst(1)
      rst.MoveNext
   Loop
End Sub
```

يفتح هذا الإجراء مجموعة سجلات لها نوع الجدول على جدول Categories باستخدام وظيفة Current DB لتمثل قاعدة البيانات المفتوحة في Access Window.

قم بإدراج إجراء Table Recordset في الوحدة النمطية bas Recordset وقم بتشغيلها في Immediate Window. يقوم الإجراء بطباعة حقل Category Name وcategory ID.

فتح مجموعة سجلات Snapshot-Type

يقوم إجراء SnapshotRecordset بتعريف db كمتغير كائن، ويشير المتغير إلى قاعدة البيانات الحالية باستخدام وظيفة Current DB. وبعد ذلك افتح مجموعة سجلات Snapshot-Type للمستخدمين من الأرجنتين.

```
Public Sub SnapshotRecordset()
   Dim db As Database
   Dim rst As Recordset
   Dim strSQL As String
   strSQL = "SELECT * FROM Customers WHERE Country = 'Argentina'"
   Set db = CurrentDB
```

```
    Set rst = db.OpenRecordset(strSQL, dbOpenSnapShot)
    Do Until rst.EOF
        Debug.Print rst("CustomerID"), rst!CompanyName
        rst.MoveNext
    Loop
End Sub
```

يستخدم الإجراء عبارة SQL كمصدر للسجلات كالتالي:

```
strSQL = "SELECT * FROM Customers WHERE Country =s 'Argentina'"
```

قم بإدخال إجراء Dynaset Recordset في الوحدة النمطية bas Recordset وقم بتشغيلها في Immediate Window. يقوم الإجراء بطباعة حقول Customer ID وcompany Name إلى Immediate Window.

فتح مجموعة سجلات في قاعدة بيانات أخرى

يفتح إجراء Recordset Other Database مجموعة سجلات لـــها نـــوع الجـــدول. لجـــدول Expense Categories في قاعدة البيانات Expenses.

```
Public Sub RecordsetOtherDatabase()
    Dim db As Database
    Dim rst As Recordset
    Set db =
DBEngine(0).OpenDatabase("c:\VBAHandbook\expenses.mdb")
    Set rst = db.OpenRecordset("Expense Categories", dbOpenTable)
    Do Until rst.EOF
        Debug.Print rst(0), rst(1)
        rst.MoveNext
    Loop
End Sub
```

عندما كان الجدول الذي تريد أن تعمل معه قاعدة بيانات أخرى، فتحتاج إلى قـــاعدة بيانـــات أخرى في الذاكرة أولاً وبعد ذلك افتح مجموعة السجلات على الجدول. ولكي تفتح قاعدة البيانات الأخرى، استخدم طريقة Open Databas لكائن Workspace. يســـتخدم الإجــراء المرجـــع الافتراضي لمسافة العمل المفتوحة حالياً (0) DB Engine. فإذا كانت قاعدة البيانات الأخرى في مجلد مختلف عن قاعدة البيانات المفتوحة حالياً في Access Window، قم بتضمين المسار فـــي وسائط لطريقة Open Database. "ففي بعض الحالات قد تحتاج إلى أن تستخدم المسار الكامل

حتى إذا كانت Expenses.mdb الموجودة في نفس المجلد مثل قاعدة البيانات الحاليـــة "يفتـــح الإجراء مجموعة سجلات لها نوع الجدول على جدول Expenses Categories وتطبـــع قيـــم الحقلين الأولئل.

قم بإدراج إجراء Recordset Other Database في الوحدة النمطيـــة Recordsets bas وقم بتشغيل الإجراء في Immdeiate Window. يوضح الشكل ١١-١٢ نتائج تشغيل إجراءات Table Recordset, Dynaset Recordset, Recordset Other Database.

الشكل ١١-١٢

نتائج الثلاث إجراءات التي تقوم بفتح مجاميع السجلات على شكل حلقات خلال السجلات لتطبع القيم من كل سجل.

```
Immediate
TableRecordset
 1          Beverages
 2          Condiments
 3          Confections
 4          Dairy Products
 5          Grains/Cereals
 6          Meat/Poultry
 7          Produce
 8          Seafood
DynasetRecordset
CACTU       Cactus Comidas para llevar
OCEAN       Océano Atlántico Ltda.
RANCH       Rancho grande
RecordsetOtherDatabase
 1          Meals
 2          Transportation
 3          Lodging
 4          Entertainment
 5          Miscellaneous
```

فتح مجموعة سجلات مستندة على كائن أخر

يمكن أن تنشئ أيضاً كائن Recordset مستند على كائن Recordset أخر، TableDef موجود، أو كائن QueryDef موجود ولكل من كائنات QueryDef وTableDef وRecordset طـــرق Open Recordset الخاص بهم. استخدم العبارات التالية لتفتح مجموعة سجلات جديدة لواحـــد من تلك الكائنات:

```
Dim rst As Recordset
Set rst = object.OpenRecordset(type, options, lockedits)
```

يعد Object كائن Recordste أو QueryDef أو TableDef موجود. وتحدد وسائط Type, Options, Lookedits الاختيارية ميزات مجموعة السجلات.

يستخدم إجراء TableDef Recordset وظيفة Current DB ليشير إلـــى قـــاعدة البيانـــات الحالية وبعد ذلك يستخدم طريقة Open Recordset لكائن TableDef لفتح مجموعة ســـجلات على الجدول.

```
Public Sub TableDefRecordset()
   Dim rst As Recordset
   Set rst = CurrentDB.TableDefs("Customers").OpenRecordset
   Do Until rst.EOF
      Debug.Print rst!CompanyName
      rst.MoveNext
   Loop
End Sub
```

افتح مجموعة سجلات جديدة على جدول Customers. قم بإدراج إجراء TableDef Recordset في الوحدة النمطية bas Recordset. وباعتباره جدول موجود، فإن Customers يعد كائن TableDef في تجمع TableDef في قاعدة البيانات الحالية. قم بتشغيل الإجراء في Immediate Window.

نقل مجاميع السجلات

عندما تقوم بإنشاء كائن Recordset، فإنك تضع صفوف من البيانات في المخزن المؤقت بالذاكرة وتشير إلى سجل فردي في وقت ما، وهو السجل الحالي. يمكن أن تعمل فقط مع السجل الحالي. انتقل خلال مجموعة السجلات عن طريق نقل مؤشر السجل الحالي إلى سجل عقب الأخر.

تم وصف التقنيات الخاصة بانتقال مجاميع السجلات في الفصل ٦، ٩. يوجد هنا recap:

- استخدم طرق ...Move وطريقة Move الخاصة بكائن Recordset لتنقل مؤشر السجل الحالي حيى تشير إلى حتى يشير إلى سجل أخر يمكن أن تستخدم خصائص BOF,EOF لتحديد إذا كنت قد نقلت مؤشر السجل الحالي خلف حدود مجموعة السجلات.
- ترجع خاصية Value لحقل في مجموعة السجلات قيمة البيانات للحقل. يمكن أن تستخدم أياً من تلك المراجع الأربع لاسترداد قيمة بيانات:

نقطة علامة تعجب	Recordsetname!Fieldname
فهرس بالاسم	Recordsetname["Fieldname"]
فهرس بمتغير	StrField="Fieldname"recordsetname[strField]
فهرس بالرقم	Recordsetname[indexnumber]

- يوجد لمجاميع السجلات طريقة Requery التي يمكن أن تستخدمها لتعيد تنفيذ الاستعلام أو إعادة قراءة الجدول المستند عليه مجموعة السجلات. فعندما تستخدم طريقة Requery، يصبح السجل الأول في مجموعة السجلات سجل حالي. فإذا كان من الممكن أن تقوم بتحديث مجموعة سجلات موجودة يعتمد على عدد من العوامل، بما في ذلك الاختيارات التي استخدمتها عندما قمت بفتحه. يمكن أن تحدد إذا كنت ستستخدم طريقة Requery عن طريق اختبار خاصية Restartable الخاصة بمجموعة السجلات فإذا كانت خاصية Restartable لمجموعة السجلات هي True، فيمكن أن تستخدم طريقة Requery لتحديث مجموعة السجلات كالأتي:

```
If rst.Restartable = True Then rst.Requery
```

ملاحظة إذا كانت خاصية Restartable لمجموعة السجلات هي False، إذاً فأنت تحتاج أن تستخدم طريقة Open Recordset لتقوم بإنشاء مجموعة سجلات جديدة لكي تقوم بالتحديث. تعد خاصية Restatable الخاصة بمجموعة السجلات التي لها نوع الجدول دائماً False، فإذا حاولت استخدام طريقة Requery على مجموعة السجلات التي لها النوع الجدول فسيرتكب VBA خطأ وقت التشغيل.

- يمكن أن تستخدم بنية Do.....Loop لتقوم بحلقات خلال كل سجل في مجموعة سجلات، كالأتي:

```
Do Until rst.EOF
    [statements]
    rst.MoveNext
Loop
```

- عندما ينتهي الإجراء، يتوقف متغير مجموعة السجلات عن التواجد ويتم غلق كائن Recordset نفسه إلا إذا كانت مجموعة السجلات هي مجموعة سجلات لنموذج يقوم بالفتح. ففي هذه الحالة، يظل كائن Recordset الخاص بالنموذج مفتوحاً بعد أن ينتهي الإجراء ويغلق فقط عندما يغلق النموذج. طبيعياً، ليس من الضروري إغلاق بوضوح كائن Recordset. ولكن إذا انتهيت منه قبل نهاية الإجراء وتريد أن تعيد استدعاء الذاكرة، فيمكن أن تستخدم طريقة Close لإغلاق مجموعة السجلات: rst.Close.
- يمكن أن تستخدم خاصية Percent Position لتحديد موقع السجل الحالي كنسبة مئوية للسجلات التي تم الوصول إليها أثناء قراءتك للخاصية. ولكي تسند Percent Position

للعدد الكلي للسجلات، يجب أن تتأكد من أن كل السجلات قد تم الوصول إليـــها، علـــى سبيل المثال، عن طريق استخدام طريقة Move Last قبل قراءة Percent Position.

تصف المقاطع التالية زوج من تقنيات تنقل مجموعة السجلات وتعطي مثـــال لتفســـير تنقـــل مجموعة السجلات.

حساب السجلات

استخدم خاصية Record Count لمجموعة السجلات لتحديد عدد السجلات. تعتمد القيمة التي تـم إرجاعها عن طريق الخاصية على نوع مجموعة السجلات. ولمجموعة السجلات التي لها نـــوع الجدول، تعطي خاصية Record Count العدد الكلي للسجلات الموجودة في الجدول. أما بالنسبة لمجاميع سجلات Dynaset-type, Snapshot-type, Forward-only-type، ترجع خاصيـــة Record Count عدد السجلات التي تم الوصول إليها.

لا يتم إرجاع خاصية Record Count إلى العدد الكلي للسجلات إلا إذا تم الوصول إلى جميـــع السجلات.

فإذا لم تقم بإلغاء السجلات، فسيجبر استخدام طريقة Move Last على التوصل إلى الســـجل الأخير، وإرجاع العدد الكلي للسجلات. فعن طريق rst المعرفة كمتغـــير لمجموعـــة الســـجلات و num المعرفة Integer، ترجع العبارات التالية العدد الكلي للسجلات:

```
rst.MoveLast
num = rst.RecordCount
```

فإذا قام الآخرين بإضافة أو إلغاء السجلات، إذاً فستحتاج إلى تحديث مجموعة الســـجلات أولاً "فرضاً أنه تم تحديث مجموعة السجلات" كالآتي:

```
rst.Requery
rst.MoveLast
num = rst.RecordCount
```

الاختبار للبحث عن مجموعة سجلات فارغة

يعد من المهم تضمين اختبار السجلات الفارغة في أي إجراء سيفشل إذا لـــم يكـــن بمجموعـــة السجلات أي سجلات. فإذا كان rst.Record Count =0، فلا يوجد أي سجلات. يوضح جـــزء الشفرة التالي اختبار بسيط يمكن أن تقوم بتضمينه فوراً عقب إنشاء مجموعة سجلات جديدة.

```
If rst.RecordCount = 0 Then
   MsgBox "There are no records!"
   rst.Close
```

```
    Exit Sub
  End If
  rst.MoveFirst
```

تعرض هذه الشفرة رسالة وتوجد من الإجراء إذا كانت مجموعة السجلات فارغة. وإلا فستنقل مؤشر السجل الحالي إلى السجل الأول.

اختبار إجراء تنقل مجموعة سجلات

يفتح إجراء Recordset Navigation مجموعة سجلات Snapshot-type على جدول Customers ويعرض موقع السجل الحالي. "عندما تفتح أولاً مجموعة سجلات يعد السجل الحالي هو السجل الأول".

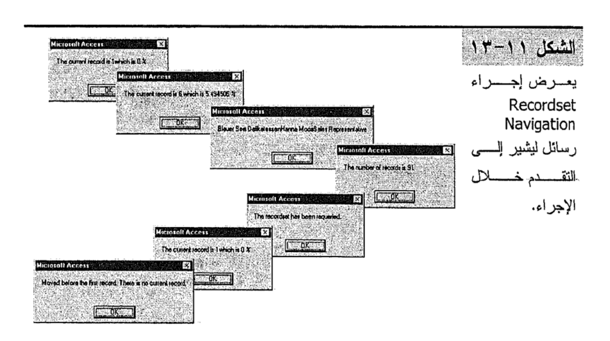
```
Public Sub RecordsetNavigation()
  Dim rst As Recordset
  Set rst = CurrentDB.OpenRecordset("Customers", dbOpenSnapshot)
  MsgBox "The current record is " & rst.AbsolutePosition + 1 & _
    " which is " & rst.PercentPosition & " % "
  rst.MoveLast
  rst.MoveFirst
  rst.Move 5
  MsgBox "The current record is " & rst.AbsolutePosition + 1 & _
    " which is " & rst.PercentPosition & " % "
  MsgBox rst!CompanyName & rst("ContactName") & rst(3)
  MsgBox "The number of records is " & rst.RecordCount
'Can you use the Requery method?
  If rst.Restartable Then
    rst.Requery
    MsgBox "The recordset has been requeried."
    MsgBox "The current record is " & rst.AbsolutePosition + 1 & _
      " which is " & rst.PercentPosition & " % "
  Else
    MsgBox "Can't requery the recordset."
  End If
  rst.MovePrevious
  If rst.BOF Then
```

```
        MsgBox "Moved before the first record. There is no current " _
          & "record."
    End If
    rst.Close
End Sub
```

ينقل الإجراء مؤشر السجل الحالي إلى نهاية مجموعة السجلات، ويرجع إلى البداية، وبعد ذلك يمضي للأمام لخمس سجلات. تعرض العبارة التالية موقع السجل الحالي، يتم البيانات في الثلاثة حقول باستخدام بناء جملة لمرجع مختلف، والعدد الكلي للسجلات.

تحدد العبارة التالية هل تسمح مجموعة السجلات بطريقة Requery أم لا. فإذا سمحت مجموعة السجلات، يقوم الإجراء بتشغيل طريقة Requery ليقوم بتحديث مجموعة السجلات وليعرض موقع السجل الحالي. تنقل العبارة التالية مؤشر السجل الحالي للخلف بواسطة سجل واحد ويختبر خاصة BOF. أخيراً، يغلق الإجراء مجموعة السجلات.

قم بإدراج إجراء Recordset Navigation في الوحدة النمطية basRecordset وبعد ذلك قم بتشغيل الإجراء في Immediate Window. يوضح الشكل "١١-١٣" الرسائل التي تشير إلى التقدم خلال الإجراء.



الشكل ١١-١٣

يعرض إجراء Recordset Navigation رسائل ليشير إلى التقدم خلال الإجراء.

إيجاد سجل محدد

وكما هو مفسر في الفصل ٦، ستجد سجل محدد في مجموعة سجلات عن طريق تحديد شريط للبحث بأنك تريد سجل لتقوم بإرضائه واستخدام واحد من التقنيات التي يوفرها محرك قاعدة

البيانات لنقل مؤشر السجل الحالي إلى السجل "الأول" الذي يرضي الشرط. استخدم تقنيات مختلفة لتجد السجلات، معتمداً على نوع مجموعة السجلات التي تقوم بإنشائها:

- تستخدم طرق Find مع مجاميع سجلات dynaset-type وsnapshot-type.
- تستخدم طريقة Seek مع مجاميع سجلات table-type.

بالإضافة إلى تقنيات كائن تشغيل البيانات، يمكن أن تستخدم طريقة Open Recordset مباشرة لتجد السجلات التي ترضي شريط البحث كما يلي، استخدم وسيط المصدر الخاص بطريقة Open Recordset لتجدد عبارة SQL بدلاً من اسم الجدول أو الاستعلام وقم بتضمين شريط البحث كجزء من عبارة SQL. فعندما تقوم بتشغيل طريقة Open Recordset، تقوم VBA بإنشاء كائن Recordset جديد يحتوي فقط على السجلات التي ترضي شريط البحث.

توفر المقاطع التالية تفاصيل وأمثلة أكثر لاستخدام طرق ...Find وSeek بالإضافة إلى تقنيات SQL لإيجاد السجل. قم بإنشاء وحدة نمطية جديدة مسماة basFinding Records لأمثلة إيجاد السجلات.

استخدام طرق Find

يعتمد أياً من الأربع طرق ....Find التي تستخدمها على المكان الذي تريد أن تبدأ منه بحثك والاتجاه التي تريد أن تبحث فيه:

- يبدأ Find First عند بداية مجموعة السجلات ويبحث في الأسفل.
- يبدأ Find Last عند نهاية مجموعة السجلات ويبحث بأعلى.
- يبدأ Find Next عند السجل الحالي ويبحث بالأسفل.
- يبدأ Find Previous عند السجل الحالي ويبحث بالأعلى.

يعد بناء الجملة متشابه لكل الطرق الأربعة. على سبيل المثال، يعد بناء الجملة لطريقة Find First هو:

```
recordset.FindFirst criteria
```

يعد Recordset اسم مجموعة السجلات الموجودة، وتعد Criteria تعبير تسلسلي مستخدم في تحديد مكان السجل. توجد هنا بعض أمثلة تعبيرات Criteria:

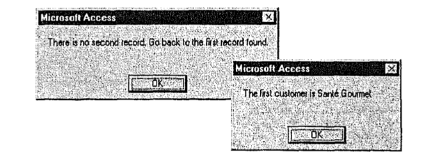
```
"OrderDate > #5-30-99# And RequiredDate <#11-30-99#"
"Country = 'Germany'"
"CompanyName Like 'B*'"
```

يستخدم إجراء Find Record كما هو واضح بالأسفل طرق Find First وfind Next ليحدد مكان السجلات.

```
Public Sub FindRecord()
   Dim db As Database
   Dim rst As Recordset
   Dim strFound As String, strCriteria as String
   Set db = CurrentDB
   Set rst = db.OpenRecordset("Customers",dbOpenDynaset)
   strCriteria = "Country = 'Norway'"
   rst.FindFirst strCriteria
   strFound = rst.Bookmark
   rst.FindNext strCriteria
   If rst.NoMatch Then
      MsgBox "There is no second record. Go back to the first " _
         & "record found."
      rst.Bookmark = strFound
      MsgBox "The first customer is " & rst!CompanyName
   Else
      MsgBox "The second customer is " & rst("CompanyName")
   End If
   rst.Close
End Sub
```

يقوم الإجراء بإنشاء مجموعة سجلات dynaset-type على جدول Customers، ويستخدم طريقة Find First لتجد المستخدم الأول من Norway، وقم بتعيين إشارة مرجعية للسجل الموجود. يستخدم الإجراء طريقة Find Next ليجد المستخدم التالي من Norway. وعندما لا يوجد مستخدم ثاني، يعرض الإجراء رسالة، ويرجع المؤشر إلى المستخدم الأول، ويعرض اسم الشركة، وإلا يعرض الإجراء رسالة مع اسم الشركة للمستخدم الثاني.

قم بإضافة إجراء Find Record والوحدة النمطية basFinding Records وقم بتشغيل الإجراء في Immediate Window. يجد الإجراء مستخدم فردي من Norway. يوضح الشكل "١١-١٤" الرسالة المعروضة.



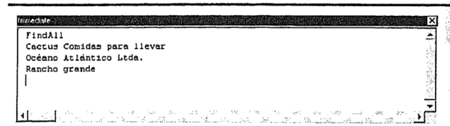
الشكل ١١-١٤
الرسالة المعروضة بواسطة إجراء Find Record.

يستخدم الإجراء Find All، الموضح بالأسفل طرق Find First وFind Next وبنية Do.......Loop.

```
Public Sub FindAll()
   Dim db As Database
   Dim rst As Recordset
   Dim strCriteria As String
   Set db = CurrentDB
   Set rst = db.OpenRecordset("Customers",dbOpenDynaset)
   strCriteria = "Country = 'Argentina'"
   rst.FindFirst strCriteria
   If rst.NoMatch Then
      MsgBox "There are no customers from Argentina."
   Else
      Do Until rst.NoMatch
         Debug.Print rst("CompanyName")
         rst.FindNext strCriteria
      Loop
   End If
End Sub
```

يستخدم هذا الإجراء طريقة Find First لتحديد مكان المستخدم الأول من Argentina. إذا لم يوجد مستخدم فسيعرض الإجراء رسالة وينتهي. وإذا كان هناك مستخدم، فسيستخدم الإجراء طريقة Find Next في بنية Do......Loop ليجد جميع السجلات الإضافية. تستخدم بنية Do.....Loop خاصية No Match كشرط حلقي، تستمر الحلقة في التنفيذ إلى أن تفشل طريقة Find Next في إيجاد سجل وتكون خاصية No Match هي True. يعرض كل تمرير للحلقة اسم الشركة للسجل الحالي ويجد السجل التالي الذي يرضي معيار البحث.

قم بإدخال إجراء Find All في الوحدة النمطية basFinding Records وقم بتشغيل الإجراء في Immediate Window. يتم طباعة أسماء الثلاث مستخدمين من الأرجنتين من أجل Immediate Window. "راجع الشكل ١١-١٥".



الشكل ١١-١٥

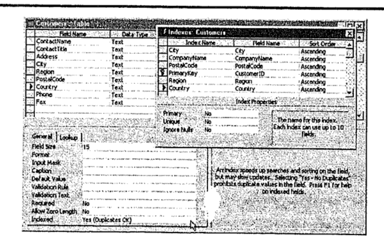
يستخدم إجراء Find All مع حلقة طريقة Find Next ليجد جميع السجلات التي ترضي معيار البحث.

استخدام طريقة Seek لمجموعة سجلات Table-type

عندما تكون مجموعة السجلات هي مجموعة سجلات Table-type لا تطبق طرق Find.....، ويجب أن تستخدم طريق Seek. تعمل Seek فقط عندما تبحث في حقل جدول يتم فهرسته. يجب أن تعين الفهرس في جدول عرض Design عن طريق إعداد خاصية Indexed الخاصة بحقل الجدول إلى [No Duplicates] Yes أولاً [Duplicates Ok] أو قم بإنشاء الفهرس للحقل كجزء من إجراء VBA. "راجع الفصل ١٤ للمزيد من المعلومات عن إنشاء فهرس برمجياً". يستخدم Seek الفهرس ليقوم بالبحث، لذلك، يعد بحث Seek أسرع من بحث Find.

ولكي تستخدم تقنية Seek لتجد المستخدم الأول من Argentina، قم أولاً بإنشاء فهرس لحقل Country عن طريق إسناد خاصية Index للحقل إلى [Duplicates Ok] Yes في جدول عرض Design "راجع الشكل ١١-١٦". ففي الإجراء قم بإنشاء مجاميع سجلات Table-type على جدول Customers وبعد ذلك استخدم العبارات الموجودة بالأسفل لتسند الفهرس الحالي إلى فهرس Country وقم بتشغيل طريقة Seek.

```
rst.Index = "Country"
rst.Seek "=", "Argentina"
```



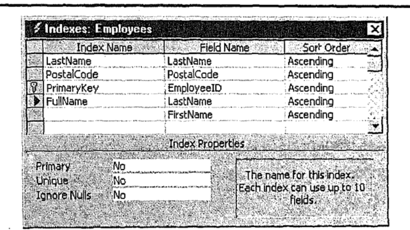
الشكل ١١-١٦

يمكن أن تقوم بإنشاء فهرس في جدول عرض Design.

ملاحظة لا توجد أي طريقة Seek Next، توجد طريقة Seek فقط السجل الأول الذي يقوم بإرضاء معيار البحث.

ولكي تستخدم تقنية Seek لتجد الموظف الأول مع اسم أول وأخير محدد، يمكن أن تقوم بإنشاء فهرس حقل متعدد في جدول عرض Design. يوضح الشكل "١١-١٧" إطار Indexes مع فهرس Full Name الذي له حقول First Name وLast Name. ففي الإجراء قم بإنشاء مجموعة سجلات Table-type على جدول Employee وبعد ذلك استخدم العبارات الموجودة بالأسفل لتسند الفهرس الحالي إلى فهرس Full Name وقم بتشغيل طريقة Seek.

```
.Index = "FullName"
rst.Seek "=", "Peacock", "Margaret"
```



الشكل ١١-١٧ يمكن أن تنشئ فهرس متعدد لحقول البحث في حوار Indexes الخاص بجدول عرض Design.

ولكي تستخدم تقنية Seek على فهرس المفتاح الأول، يمكن ان تعرف الفهرس مع السلسلة "Primary Key". على سبيل المثال، لكي تستخدم تقنية Seek لتجد الترتيب الأول مع Order ID أكبر من ١١٠٤٠، قم بإنشاء مجموعة سجلات Table-type على جدول Orders وبعد ذلك استخدم العبارات الموجودة بالأسفل لتسند الفهرس الحالي إلى فهرس المفتاح الأول وقم بتشغيل طريقة Seek.

```
rst.Index = "PrimaryKey"
rst.Seek ">", 11040
```

ففي المثال الأول بأعلى، يبحث Access عن فهرس Country عن المرة الأولى التي تظهر بها Argentina في جدول بحث الفهرس وبعد ذلك استخدم جدول بحث الفهرس لتحدد مكان سجل الجدول المماثل واجعله السجل الحالي. وإذا لم يتم إيجاد أي سجل، يعد مؤشر السجل الحالي في

Limbo، ويعد السجل الحالي غير معرف، استخدم خاصية No Match لتحدد إذا كانت تقنية Seek ناجحة أم لا. وتوضح هذه التقنية إجراء Seek All الواضح بالأسفل.

```
Public Sub SeekAll()
   Dim rst As Recordset
   Set rst = CurrentDB.OpenRecordset("Customers")
   rst.Index = "Country"
   rst.Seek "=", "Argentina"
   If rst.NoMatch Then
      MsgBox "There are no customers from Argentina."
   Else
      MsgBox "The first customer from Argentina is " & _
         rst!CompanyName
   End If
End Sub
```

يفتح هذا الإجراء مجموعة سجلات على جدول Customers. "فعندما تفتح مجموعة سجلات على جدول في قاعدة البيانات الحالية بدون تحديد نوع مجموعة السجلات، تقوم Access بإنشاء مجموعة سجلات Table-type". تسند العبارات التالية الفهرس الحالي إلى حقل [Country] المفهرس وابحث في الفهرس عن المستخدم الأول من Argentina.

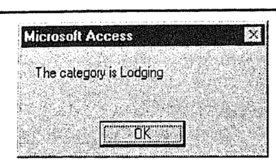
افتح جدول Customers في عرض Design، وقم بإسناد خاصية Indexed لحقل Country في جدول Customers إلى [Duplicates Ok] Yes. احفظ الجدول. بعد ذلك، قم بإدراج إجراء Seek All في الوحدة النمطية basFinding Records وقم بتشغيله في Immediate Window. يستخدم الإجراء خاصية No Match ليحدد إذا كان البحث ناجح ويطبع نتائج الاختبار أم لا.

يمكن فتح مجموعة سجلات Table-type فقط على جدول في قاعدة بيانات مفتوحة. لذلك، لا يمكن ان تفتح مجموعة سجلات Table-type على جدول مرتبط في قاعدة بيانات mdb. أخرى من قاعدة البيانات الحالية أو على جدول ODBC مرتبط "مثل جدول Excel المرفق". بينما يمكن أن تستخدم تقنية Seek في جدول في قاعدة بيانات mdb. أخرى، فهل تم ربط الجدول أم لا، عن طريق فتح أولاً قاعدة البيانات الأخرى في الإجراء. بمجرد فتح قاعدة البيانات، يمكن أن تستخدم طريقة Open Recodset لقاعدة البيانات تلك لتفتح مجموعة سجلات Table-type على واحد من جداوله. وعلى سبيل المثال، يمكن أن نستخدم طريقة Seek للبحث عن جدول في قاعدة بيانات Expenses "تم إنشائها في الفصل ١". ويفسر هذه التقنية إجراء Seek Other Database.

```
Public Sub SeekOtherDatabase
   Dim ws As Workspace
   Dim db As Database, rst As Recordset
   Set ws = DBEngine(0)
   Set db = ws.OpenDatabase("c:\VBAHandbook\expenses.mdb")
   Set rst = db.OpenRecordset("Expense Categories")
   rst.Index = "Primarykey"
   rst.Seek "=", 3
   If rst.NoMatch Then
      MsgBox "There is no such category."
   Else
      MsgBox "The category is " & rst!ExpenseCategory
   End If
End Sub
```

يفتح الإجراء قاعدة بيانات Expenses باستخدام طريقة Open Database الخاصة بكائن Workspace. "يمكن أن تحتاج إلى تضمين معلومة المسار، وخاصة إذا لم يكن ملف قاعدة البيانات في المجلد الذي يحتوي على قاعدة البيانات التي تم فتحها حالياً في Access Window". فبعد فتح قاعدة البيانات Expenses في الذاكرة، يستخدم الإجراء طريقة Open Recordset على كائن قاعدة البيانات Expenses لفتح مجموعة سجلات Table-type على جدول Expense Categories. يعد Expense Category ID المفتاح الأول لهذا الجدول. يسند الإجراء الفهرس الحالي إلى فهرس المفتاح الأول وبعد ذلك استخدم طريقة Seek لتجد السجل مع ID of 3. تحدد العبارات القادمة إذا كان البحث ناجح وتطبع نتيجة البحث في Immediate Window أم لا.

قم بإدراج إجراء Seek Other Database في الوحدة النمطية basFinding Records وقم بتشغيله في Immediate Window، يعد البحث ناجحاً "راجع الشكل ١١-١٨".



الشكل ١١-١٨
نتيجة بحث الجدول في قاعدة بيانات أخرى .mdb باستخدام طريقة .Seek

استخدام سلسلة SQL مع معيار البحث

غالباً، تعد أسرع طريقة لتجد سجلات هي استخدام تقنيات SQL بدلاً من طرق Seek أو Find... ولكي تجد سجل يستخدم SQL، تفتح ببساطة مجموعة السجلات باستخدام عبارة SQL لتحدد مجموعة السجلات كوسيط طريقة Open Recordset بدلاً من اسم الجدول أو الاستعلام. على سبيل المثال، تعد عبارة SQL لإيجاد المستخدمين من Argentina هي:

```
SELECT * FROM Customers WHERE Country = "Argentina"
```

إذا لم تعرف SQL، يمكن أن تستخدم استعلام عرض Design لإنشاء الاستعلام، قم بالتبديل إلى عرض SQL، انسخ عبارة SQL المعادلة، والصق النتيجة داخل شفرتك، يعد اصعب جزء هو تحويل عبارة SQL إلى تسلسل يقبله محرك قاعدة البيانات. فعندما تحتوي عبارة SQL على تسلسل كما في هذا المثال، يعد وسيط المصدر لطريقة Open Recordset تعبير تسلسلي يحتوي على تسلسل. يجب أن تعرف التسلسل الداخلي داخل التعبير التسلسلي الخارجي برموز غير علامات الترقيم المضاعفة، لأن علامات الترقيم المضاعفة قد تم استخدامها بالفعل للإشارة إلى التغيير التسلسلي الخارجي. وتعمل أياً من التعبيرات الآتية:

```
strSQL = "SELECT * FROM Customers WHERE Country = ""Argentina"""
strSQL = "SELECT * FROM Customers WHERE Country = 'Argentina'"
```

يفتح إجراء SQL Records مجموعة سجلات تحتوي فقط على المستخدمين من Argentina وتستخدم بنية Do......Loop لطباعة أسماء الشركة في Immediate Window.

```
Public Sub SQLRecords()
    Dim rst As Recordset
    Dim strSQL As String
    strSQL = "SELECT * FROM Customers WHERE Country = 'Argentina'"
    Set rst = CurrentDB.OpenRecordset(strSQL)
    Do Until rst.EOF
        Debug.Print rst!CompanyName
```

```
        rst.MoveNext
    Loop
End Sub
```

ادخل إجراء SQL Records في الوحدة النمطية basFinding Records وقم بتشغيله في Immediate Window. لقد تم مناقشة استخدام تقنيات SQL بتفصيل أكثر في الفصل ١٣.

تقرير تقنية بحث تستخدمها

تعتمد تقنية البحث التي يجب أن تستخدمها على عوامل متعددة. فإذا احتجت فقط إلى السجلات التي ترضي شريط بحث، فغالباً تعطي تقنية SQL الأداء الأفضل، بينما، إذا احتجت إلى جميع السجلات التي تم إرجاعها بواسطة جدول أو استعلام، سواء يقومون بإرضاء شريط بحث أم لا، بعد ذلك ستحتاج أن تنشئ مجاميع سجلات لترجع جميع السجلات بأي طريقة، قد يعطي استخدام تقنيات Seek أو ....Find لتحديد مكان السجل المحدد أفضل أداء شامل.

فإذا كنت قد حددت بالفعل مكان سجل محدد عن طريق أياً من التقنيات وخططت للرجوع إلى السجل مؤخراً في الإجراء، فتعد أسرع طريقة للرجوع إلى السجل هي استخدام إشارة مرجعية.

ولكي تجد سجل للمرة الأولى، تعد القاعدة العامة هي أن طريقة Seek الأسرع "ولكن تعد محدودة على مجاميع سجلات Table-type"، تعد طريقة SQL هي التالية، وتعد طرق ....Find هي الأبطأ. فكلما زادت السجلات التي يجب أن تبحثها، كلما كبر اختلاف الأداء بين طرق .....Find وSQL.

> **ملاحظة** عندما تقرر تقنية البحث التي تعطي أفضل أداء، استخدم وظيفة Timer لتحسب الوقت ولتقارن الإجراءات. راجع الفصل ٩ للمزيد من المعلومات عن استخدام وظيفة Timer.

استكشاف Clones

وكما هو مفسر في الفصل ٦، يمكن أن تستخدم طريقة Clone الخاصة بكائن Recordset لتقوم بإنشاء كائن Recordset جديد مطابقاً لكائن Recordset الأصلي باختلاف هام: فقد سمي كائن Recordset الجديد Clone الخاص بالأصل، وله مؤشر سجل حالي مستقل خاص به. يعد إنشاء Clone أسرع من إنشاء كائن Recordset جديد باستخدام Open Recordset.

ملاحظـــة عندما تقوم بإنشاء clone لكائن Recordset موجـــود باســـتخدام طريقـــة Clone، يكون لكل مـــن كائنـــات Recordset الأصليـــة وclone نفـــس الإشارات المرجعية. ويوجد لأثنين من كائنات Recordset التي تم إنشاءها عن طريق أياً من الوسائل الأخرى مجاميع إشارات مرجعية مختلفة، حتـــى عندما يكونوا مستندين على نفس الجدول أو الاستعلام أو العرض أو عبـــارة SQL. لا تستطيع أن تقوم بتزامن سجلاتهم الحاليـــة باســـتخدام إشـــارات مرجعية. على سبيل المثال، إذا قمت بفتح نموذج مستند على جدول وبعـــد ذلك تستخدم طريقة Open Recordset لتفتح مجموعة سجلات على نفـــس الجدول، يكون لأثنين من كائنات مجموعة السجلات مجموعات منفصلة من الإرشادات المرجعية.

وكمثال لاستخدام Clone، يقارن إجراء Duplicates القيم الموجودة فـــي حقـــل Country لسجلين متتاليين للمستخدم.

```
Public Sub Duplicates()
    Dim db As Database, rst As Recordset
    Dim rstClone As Recordset
    Set db = CurrentDB
    Set rst = db.OpenRecordset("Customers")
    rst.Move 20
    Set rstClone = rst.Clone
    rstClone.Bookmark = rst.Bookmark
    rstClone.MovePrevious
    If rstClone!Country = rst!Country Then
        MsgBox "The previous record has the same value for Country."
    Else
        MsgBox " The previous record does not have the same value _
            for Country."
    End If
    MsgBox "Previous record value: " & rstClone!Country _
    & "Current record value: " & rst!Country
    rst.Close
    rstClone.Close
End Sub
```

يفتح هذا الإجراء مجموعة سجلات Table-type على جدول Customers وينقل مؤشر السجل الحالي ٢٠ صفاً للأمام للسجل الحادي والعشرين. يقوم الإجراء بإنشاء Clone ويتزامن مع مؤشر السجل الحالي الذي يخص Clone مع نفس السجل. تنقل طريقة Move Previous مؤشر السجل الحالي الذي يخص Clone إلى السجل السابق الذي يخص Clone. يقارن الإجراء القيم الموجودة في حقل Country للأصل "سجل ٢١" وClone "سجل ٢٠" ويعرض نتائج المقارنة والقيم الموجودة في السجلين. تغلق العبارة التالية كل كائنات Recordset.

قم بإدراج إجراء Duplicates في الوحدة النمطية basFinding Records وقم بتشغيله في Immediate Window. يتم عرض القيم الموجودة في السجلين في مربع للرسالة (راجع الشكل ١١-١٩).

الشكل ١١-١٩

عندما يحتاج إجراء أن يعمل مع أكثر من سجل في وقت ما، استخدم طريقة Clone لتقوم بإنشاء مجموعة سجلات متضاعفة مع مؤشر السجل الحالي الخاص بها.

Microsoft Access

Previous record value: Austria Current record value: Brazil

OK

قراءة بيانات الجدول داخل مصفوفة

عندما لا تحتاج إلى تغيير البيانات، يعد إنشاء مجموعة سجلات النوع الوحيد للأمام هو أسرع طريق لاسترداد مجموعة من السجلات يمكن أن تستخدم فقط Move Next أو طريقة Move لتنقل مؤشر السجل الحالي للأمام. يمكن أن تستخدم فقط طرق Find Next وFind First لتجد سجلات محددة لأن تلك الطرق تنقل مؤشر السجل الحالي إلى الأمام.

فإذا لم تحتاج إلى تغيير البيانات ولكنك تحتاج إلى الوصول العشوائي للبيانات، فلم تعد مجموعة سجلات النوع الوحيد للأمام هي الحل. فبدلاً من ذلك، يمكن أن تقوم بإنشاء مصفوفة لحمل البيانات في الذاكرة "تم مناقشة المصفوفات في الفصل ٨". فعقب قراءتك للبيانات من

مجموعة السجلات داخل مصفوفة، يمكن أن تغلق مجموعة السجلات وتدع الجداول للآخرين كي يستخدمونها. "إذا كنت تعمل في بيئة متعددة المستخدمين، فسيصغر استخدام المصفوفات ملابسات إغلاق السجل".

استخدم طريقة Get Rows الخاصة بالكائن Recordset لتنسخ صفوف من كائن Recordset إلى مصفوفة متنوعة لها بعدين، ويعد بناء الجملة هو:

```
varArray = recordset.GetRows(number)
```

ففي بناء الجملة هذا يعد recordset أي نوع من مجموعة السجلات، ويعد VarArray متغير لنوع البيانات Variant، ويعد Number هو رقم الصفوف التي تريد نسخها.

تعد المصفوفة التي تم إرجاعها بواسطة طريقة Get Rows هي مصفوفة لها بعدين، ومعها العنصر الأول الذي يعرف الحقل والعنصر الثاني الذي يعرف الصف. وعلى سبيل المثال، تعد VarArray [2,3] هي قيمة الحقل الثالث في الصف الرابع "يبدأ Access كل فهرس بصفر".

فإذا طلبت صفوف أكثر من الصفوف المتوفرة يتم إرجاع الصفوف المتوفرة فقط. يمكن أن تستخدم وظيفة UBound لتحدد عدد الحقول والصفوف التي تم إرجاعها ولتحدد عدد الحقول والصفوف التي تم إرجاعها، استخدم العبارات:

```
numFields = UBound(varArray, 1) + 1
numRows = UBound(varArray, 2) + 1
```

وبما أن فهارس الحقل والصف قد بدأ بصفر، تضيف العبارات واحداً للفهارس الكبيرة للحصول على عدد الحقول والصفوف.

تنسخ طريقة Get Rows عدد السجلات المحدد بدءاً من السجل الحالي. فبعد تنفيذ طريقة Get Rows، يعد السجل الحالي هو الصف التالي الغير مقروء. تم إرجاع جميع حقول مجموعة السجلات، بما في ذلك الحقول الثنائية وحقول الذاكرة. فإذا لم تكن تريد تضمين جميع الحقول، استخدم استعلام أو عبارة SQL لتقيد الحقول في مجموعة السجلات قبل استخدام طريقة Get Rows.

ولكي تستكشف تلك المفاهيم، قم بإنشاء وحدة نمطية جديدة مسماة basArrays وقم بتعريف متغير VarArray في مقطع Declarations كمتغير له مستوى الوحدة النمطية العامة باستخدام عبارة التعريف:

```
Public varArray As Variant
```

استخدم متغير له مستوى وحدة نمطية حتى تستمر المصفوفة في وجودها بعد قراءة القيم داخلها بواسطة إجراء Array Recordset الواضح بالأسفل.

```
Public Sub ArrayRecordset()
    Dim rst As Recordset
    Dim numFields As Integer, numRows As Integer
    Dim j As Integer
    Set rst = CurrentDB.OpenRecordset("Customers", dbOpenDynaset)
    rst.Move 20
    Debug.Print rst.AbsolutePosition
    varArray = rst.GetRows(3)
    Debug.Print rst.AbsolutePosition
    rst.Close
    numFields = UBound(varArray, 1) + 1
    numRows = UBound(varArray, 2) + 1
    Debug.Print "Fields: " & numFields & " Rows: " & numRows
    For j = 0 To numRows - 1
        Debug.Print varArray(0,j) & " Company Name: " & varArray(1,j)
    Next
End Sub
```

يفتح هذا الإجراء مجموعة سجلات dynaset-type على جدول Customers، انقل ٢٠ صف للأمام، قم بطباعة موقع الصف المطلق، اقرأ ثلاثة صفوف داخل مصفوفة متنوعة، قم بطباعة موقع الصف المطلق مرة ثانية، أغلق مجموعة السجلات. يحدد الإجراء عدد الحقول والصفوف التي تم قراءتها بالفعل داخل المصفوفة ويستخدم حلقة For.....Next ليطبع خارجياً القيم الموجودة في الحقلين الأولين لكل صف في المصفوفة.

قم بإدخال إجراء Array Recordset في الوحدة النمطية basArrays وقم بتشغيلها في Immediate Window. ولأن VarArray تعد متغير عام، يتم الاحتفاظ بقيمتها إلى أن تغلق قاعدة البيانات تعد القيم متوفرة لجميع الوحدات النمطية وأيضاً Immediate Window. اكتب؟ varArray (1,2) واضغط Enter. يتم طباعة القيم في الحقل الثاني الخاص بالصف الثالث "راجع الشكل ١١-٢٠".

```
arrayrecordset
 20
 23
Fields: 11Rows: 3
FAMIA Company Name: Familia Arquibaldo
FISSA Company Name: FISSA Fabrica Inter. Salchichas S.A.
FOLIG Company Name: Folies gourmandes
?vararray(1,2)
Folies gourmandes
```

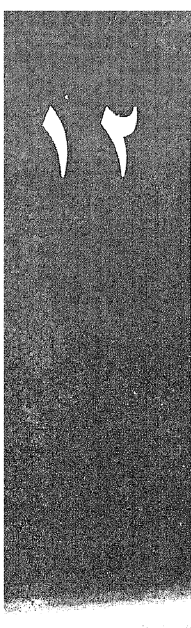
الشكل ١١-٢٠ يستخدم إجراء Array Recordset طريقة Get Rows لقراءة البيانات داخل مصفوفة.

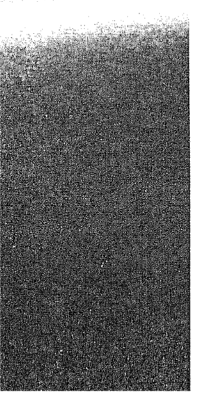
خلاصة

يركز هذا الفصل على استخدام إجراءات VBA لجعل التنقل آلياً في واجهة النماذج ولاستخدام VBA للتنقل خلال البيانات عن طريق كائنات مجموعة السجلات في الذاكرة بدون نماذج فاتحة. تأتي هنا النقاط الهامة.

- يمكن استخدام طرق كائن Docmd لتضاعف ضغط الأزرار ونقر الماوس وتحديدات أمر القائمة الخاصة بالبيئة التفاعلية.
- بينما تعد إجراءات الحدث مفيدة في إنشاء إجراءات التنقل في الواجهة، إلا أن إنشاء إجراءات قابلة لإعادة الاستخدام من الطبيعي أن تتطلب استخدام إجراءات وظيفية وتخزينها في الوحدات النمطية القياسية.
- استخدم خاصية Recordset Clone الخاصة بالنموذج لتقوم بإنشاء مرجع لكائن Recordset الخاص بالنموذج الذي مؤشر سجل حالي مستقل عن مؤشر السجل الحالي للنموذج. يعطيك استخدام Recordset Clone الوصول إلى معظم طرق وخصائص كائن Recordset.
- يتطلب إنشاء أزرار تنقل مخصصة للاستعراض خلال السجلات، إجراءات تنقل مؤشر السجل الحالي خلف حدود مجموعة السجلات. استخدم خصائص Bof وEof لتحدد إذا كنت قد تخطيت الحدود أم لا.
- يمكن أن تجد سجل محدد باستخدام نموذج بثلاثة طرق:
  - تضاعف طريقة Find Record لكائن Docmd المنهج التفاعلي لإيجاد سجل.
  - تطبيق طريقة Apply Filter لكائن Docmd تصفيه على السجلات.
  - تسمح خاصية Recordset Clone لك باستخدام طرق ....Find لكائن Recordset الخاص بالنموذج.

- يمكن أن تعمل مباشرة مع البيانات بدون فتح نموذج عن طريق إنشاء واحداً من أربع أنواع لكائنات Recordset الموجودة في الذاكرة. يمكن أن تؤسس مجموعة سجلات على جدول أو استعلام أو عرض أو عبارة SQL التي ترجع السجلات.
- يعد المفتاح لفهم مجاميع السجلات هو السجل الحالي. يمكن أن تعمل فقط مع السجل الحالي، واستخدم طرق ....Move لتنقل مؤشر السجل الحالي من سجل إلى أخر.
- ولكي تعمل مع سجلين فوراً، يمكن أن تفتح مجموعة سجلات ثانية أو استخدام طريقة Clone لتنشئ تضاعف لمجموعة سجلات الأصل ولكن عن طريق مؤشر سجل حالي مستقل.
- ولكي تجد سجل محدد باستخدام مجموعة السجلات، افعل واحدة من الأتي:
  - تحديد شريط البحث في عبارة SQL وقم بإنشاء مجموعة سجلات مستندة على عبارة SQL.
  - قم بإنشاء مجموعة سجلات وبعد ذلك استخدم طريقة Seek لمجموعة سجلات Table-type عندما يكون شريط البحث مستند على فهرس أو طرق ...Find للأنواع الأخرى لمجاميع السجلات.
- إذا لم تحتاج إلى تغيير البيانات ولكن تحتاج إلى وصول عشوائي، يمكن أن تقرأ بيانات داخل مصفوفة.





معالجة البيانات باستخدام إجراءات VBA في أكسس

يمكن استخدام طريقتان أساسيتان للعمل مع البيانات في تطبيق قاعدة البيانات وهما: طريقة النماذج وطريقة مجموعة السجلات.

بالنسبة لطريقة النماذج فهي التي تقدم من خلالها النماذج التي يعمل معها المستخدم لإضافة سجلات جديدة أو لتغيير قيم البيانات في الحقول أو لحذف سجلات موجودة بالفعل. كما أنك تستخدم إجراءات VBA لأتمتة عمليات معالجة البيانات والتي منها عرض سجل خالٍ أو إعداد قيم افتراضية للحقول في سجل جديد ة والتحقق من صحة البيانات الجديدة أو المتغيرة في الحقول والتراجع عن التغيرات أو حفظها في السجل وحذف سجل موجود بالفعل. غير أن المشكلة التي تقترن باستخدام النماذج لتغيير قيم البيانات تتمثل في عدم قدرة أكسس في بعض الأحيان على تحديث السجلات تلقائياً وعرض أكثر القيم حداثة "أنظر الفصل الرابع لمزيد من الأمثلة". وهذا الفصل يعرض كيفية استخدام إجراءات VBA لتحديث مصادر البيانات لعرض أكثر البيانات حداثة.

أما في طريقة مجموعة السجلات، فإنك تقوم بإضافة أو تغيير أو حذف سجلات دون الحاجة إلى عرض السجلات في نموذج للمستخدم. فلنفرض مثلاً أنك تريد تغيير الاسم الوظيفي لممثلي المبيعات ليصبح Account Executive، فستقوم بإنشاء إجراء في طريقة مجموعة السجلات يفتح مجموعة سجلات في جدول Employees ويحدث التغيير اللازم في الذاكرة بدون عرض السجلات للمستخدم.

معالجة البيانات باستخدام النماذج

في معظم تطبيقات قاعدة البيانات تقوم بتقديم مجموعة نماذج تسمح للمستخدم بالعمل مع البيانات والشيء الأساسي في العديد من هذه التطبيقات هو مجموعة نماذج إدخال البيانات المصممة لإدخال سجلات جديدة وتعديل السجلات الموجودة بالفعل. وغالباً تحتاج نموذج إدخال البيانات ليعمل كنموذج السجلات ، وفي وضع إدخال البيانات تكون أدوات تحكم البيانات ملغي تأمينها ومتاحة بحيث يستطيع المستخدم تغيير البيانات. أما في مراجعة الوضع فعادةً ما تكون أدوات تحكم البيانات مغلقة وغير متاحة فلا يستطيع المستخدم إحداث أية تغييرات . ومن مفاهيم التصميم الهامة هي استخدام مراجعة الوضع للمحافظة على البيانات من التغييرات غير المقصودة كما تتضمن أيضاً طريقة ما تحميها كلمة مرور يستعين بها المستخدم لتغيير النموذج إلى وضع إدخال البيانات.

ملاحظة للإلمام بالتقنيات المذكورة في هذا الفصل قم بإنشاء نسخة جديدة لنموذج Northwind لقاعدة البيانات تحمل اسم Northwind_ch12 وتعمل من خلال خطوات النماذج.

تبديل النموذج بين وضع المراجعة ووضع إدخال البيانات

كطريقة جديدة لتجنب إحداث تغييرات غير مقصودة بالبيانات افتح النموذج في وضع المراجعة واستخدم زر أمر للتغيير بين وضع المراجعة ووضع إدخال البيانات، فالنقر على زر أمر ينتج عنه تشغيل إجراء يقفز في أدوات تحكم البيانات للنموذج ويغير خصائصها Locked وEnabled "انظر الفصل ٩ لمزيد من المعلومات عن إنشاء حلقات من خلال مجموعات". فعلى سبيل المثال بالنسبة للإجراءات LockControl وUnlockControls الموضحة بأسفل استخدم التركيب For Each...Next لتنفيذ حلقة بكل أداة تحكم في النموذج إذا كان أداة التحكم هو مربع النص أو مربع تحرير وسرد فإن الإجراء سيقوم بتعيين قيمة خصائص Locked وEnabled.

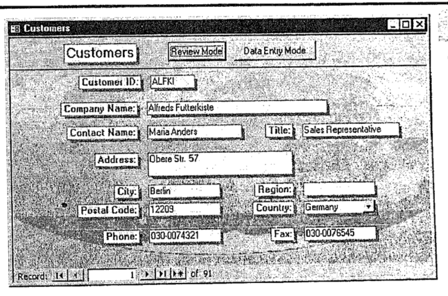
```
Public Function LockControls()
   Dim ctl As Control
   For Each ctl in Screen.ActiveForm.Controls
      If TypeOf ctl is TextBox Or TypeOf ctl is ComboBox Then
         With ctl
            .Locked = True
            .Enabled = False
         End With
      End If
   Next
End Function
Public Function UnLockControls()
   Dim ctl As Control
   For Each ctl in Screen.ActiveForm.Controls
      If TypeOf ctl is TextBox Or TypeOf ctl is ComboBox Then
          With ctl
            .Locked = False
            .Enabled = True
         End With
      End If
   Next
End Function
```

لتجهيز نموذج Customers كنموذج وضع مزدوج، افتح نموذج Customers في عرض Design. حدد جميع مربعات النص ومربع التحرير والسرد Country، وقم بـإعداد خاصية Enabled في No وخاصية Locked في Yes للتحديد المتعدد.

بعد ذلك، قم بإنشاء وحدة نمطية قياسية تحمل اسم basSpport ثم إدراج إجراءات LockControls وunlockControls المذكورة سلفاً. ضع زرين أمر في مقطع رأس الصفحة للنموذج Customers، وأعطهم خصائص Name وcaption وOn Click التالية.

اسم الزر	التعليق	إجراء الحدث
CmdLock	وضع المراجعة	=LockControls()
CmdUnLock	وضع إدخال البيانات	=UnLockControls()

احفظ النموذج وانتقل إلى عرض Form، وسيصبح النموذج بأدوات تحكم البيانات الخاصة به المغلقة "Locked" وغير المتاح "disabled" في وضع المراجعة "انظر الشكل ١٢-١". انقر Data Entry Mode وقم بتغيير البيانات في مربع النص ثم انقر زر Review Mode لإعادة غلق أدوات التحكم.



الشكل ١٢-١ تجنب إحداث تغييرات غير مقصودة بالبيانات وذلك بإنشاء إجراءات تنفذ حلقة في أدوات تحكم البيانات، وتقوم بالتبديل بين خصائص Locked وenabled لتلك البيانات.

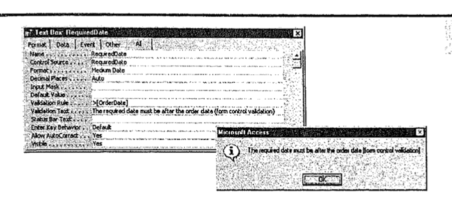
التحقق من صحة البيانات

يلعب تصميم التطبيق دوراً هاماً في تحديد كيفية تحقق التطبيق من البيانات الجديدة أو المتغيرة. ويقدم أكسس مجموعة من الخصائص التي يمكن استخدامها عند إنشاء جدول لاختبار البيانات قبل حفظها في القرص، ومن هذه الخصائص Required و AllowZeroLength و Validation Rule للحقول وخاصية Validation Rule للجدول كما يمكن إعداد خاصية Validation Rule لأدوات التحكم بالنماذج. تستخدم خصائص Validation Rule لحقل الجدول أو للجدول أو لأداة تحكم نموذج لتحديد المتطلبات التي يجب أن تفي بها البيانات، كما تستخدم خاصية Validation Text لتحديد النص الذي تريد أن يعرضه أكسس في مربع رسالة في حالة أن البيانات الجديدة أو المتغيرة ليست كافية بالنسبة لإعداد Validation Rule.

ومثال على ذلك عندما يتم وضع ترتيب جديد باستخدام نموذج Orders في قاعدة البيانات Northwind فإن تاريخ الترتيب يتم وضعه تلقائياً وذلك لأن خاصية Default Value لأداة التحكم Order Date تعد لتكون Date()= لكن المستخدم يقوم بتعبئة التاريخ المطلوب كجزء من اتخاذ الترتيب ولم يتم إعداد أي قاعدة للتحقق من الصحة لحقل Required Date. لتحديد التاريخ الموجود ليكون أكبر من تاريخ الترتيب يمكن إعداد إحدى خصائص Validation Rule حتى يستطيع أكسس اختبار الشرطRequired Date ⇦ Order Date ورفض القيام بالتحديث إذا لم يتم تحقيق الشرط.

لكن أي من خصائص Validation Rule ستستخدم ؟ فلا يمكنك تحديد الاختبار باستخدام خاصية Validation Rule لجدول أو لأداة تحكم في نموذج Orders.

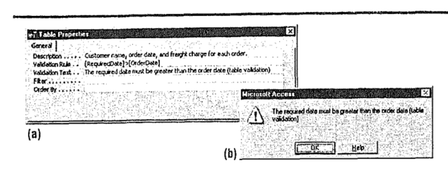
والخاصية التي تختارها تكون حسب الموعد الذي تريد فيه أكسس أن يختبر البيانات، فيختبر أكسس خاصية أداة التحكم Validation Rule عند محاولة نقل التركيز خارج أداة التحكم بعد إدخال بيانات فيختبر أكسس خاصية أداة التحكم Validation Rule عند محاولة نقل التركيز خارج أداة التحكم بعد إدخال بيانات جديدة أو تغيير الموجود منها بالفعل في أداة التحكم. ويوضح الشكل ١٢-٢أ خاصية Validation Rule لأداة التحكم Required Date كما يوضح الشكل ١٢-٢ب الرسالة التي تعرض عند إدخال تاريخ غير صالح أو عند إنشاء علامة جدولة خارج أداة التحكم.



الشكل ١٢-٢
يقوم أكسس باختبار خاصية أداة التحكم Validation Rule "أ" عند إدخال أو تغيير البيانات في أداة التحكم حاول نقل التركيز خارج أداة التحكم. إذا لم يتم تحقيق الشرط يقوم أكسس بعرض النص الذي تقوم بتحديده كخاصية Validation Text "ب"

ومن ناحية أخرى يقوم أكسس باختبار خاصية الجدول Validation Rule عند محاولة حفظ السجل بعد إدخال البيانات الجديدة أو تغيير البيانات الموجودة بالفعل. تبعاً لذلك يمكن أن يقوم المستخدم بإدخال بيانات غير صالحة في السجل ولا يدرك وجود مشكلات التحقق من الصحة إلا عند محاولة حفظ السجل. ويوضح الشكل ١٢-٣ أ خاصية Validation Rule لجدول Orders، بينما يوضح الشكل ١٢-٣ ب الرسالة التي تعرض عند إدخال تاريخ غير صالح ومحاولة حفظ السجل.

الشكل ١٢-٣



ومن ناحية أخرى يقوم أكسس باختبار خاصية الجدول Validation Rule عند محاولة حفظ السجل بعد إدخال البيانات الجديدة أو تغيير بيانات في السجل. وعند محاولة حفظ السجل "أ" إذا لم يتم تحقيق الشرط يقوم أكسس بعرض رسالة افتراضية أو بعرض النص الذي قمت بتحديده كخاصية Validation Text "ب".

ملاحظة عندما يحتوي النموذج على نموذج ثانوي فيكون نقل التركيز ليصبح هذا النموذج الثانوي محاولة لحفظ التغييرات التي أحدثتها في السجل في النموذج الأساسي كما يحدث أيضاً اختبار خاصية Validation Rule لمصدر السجل الأساسي للنموذج الأساسي.

تنفذ عملية التحقق من أدوات التحكم لنموذج أو لحقول في الجدول الأساسي بالترتيب على النحو التالي:

١- خاصية أداة التحكم Validation Rule عند محاولة تحديث أداة تحكم متغيرة.

٢- خاصية Validation Rule الأساسية للحقل عند محاولة تحديث أداة تحكم متغيرة.

٣- خاصية Validation Rule الأساسية للجدول عند محاولة حفظ سجل متغير.

عند محاولة حفظ سجل متغير، بالإضافة إلى اختبار خاصية Validation Rule للجدول، يقوم أكسس باختبار القواعد التمامية للجدول. إذا ما قمت بتعيين مفتاحاً أساسياً، فإن أكسس يقوم باختبارات لمعرفة ما إذا كانت حقول المفتاح الأساسي غير خالية وإذا كان المفتاح الأساسي منفردا "تمامية الكيان". بالإضافة إلى ذلك إذا كان الجدول طرفاً في أكثر من علاقة ثم قمت أنت بمراجعة خيار التمامية المرجعي للعلاقة، فإن أكسس يقوم بإجراء اختبارات للتأكد من أن التغييرات التي قمت بإحداثها لن تنشئ أي سجلات وحيدة.

عند استخدام خاصية Validation Rule فإنك تكون محدوداً بالطريقة الافتراضية: يقوم أكسس باختبار القاعدة وإما أن يقبل البيانات أو يرفضها ثم يعرض رسالة. وعندما تكون هذه الطريقة الافتراضية غير كافية، فإنه يمكن استخدام إجراءات VBA لإعداد قواعد اكثر تعقيداً للتحقق من الصحة، وللتحكم فيما يقوم به أكسس عند انتهاك القواعد فعلى سبيل المثال يمكن استخدام البرمجة لعمل الآتي:

- لعرض رسالات مختلفة معتمدة على القيمة التي تم إدخالها.
- لسؤال مستخدم الإدخال عن إذا ما كان يمكنه استخدام قاعدة التحقق من الصحة.
- لاستخدام أكثر من قاعدة تحقق من الصحة للتحقق من صحة السجل.
- لتنفيذ عمليات إضافية تعتمد على نتيجة اختبار التحقق من الصحة.
- لتغيير توقيت اختبارات التحقق من الصحة.

استخدام إجراء حدث "Event" لإلغاء السلوك الافتراضي

عند استخدام VBA في أكسس للتحقق من صحة البيانات، فإنك عادة تقوم بإنشاء إجراء حدث يتضمن الشروط التي تريد اختبارها والعمليات التي تريد تنفيذها وذلك عند تحقيق شرط واحد أو أكثر. ويكر استخدام الأحداث المذكورة في الجدول ١٢-١ لإحداث إجراء التحقق من صحة الحدث.

الجدول ١٢-١ : أحداث شائعة تستخدم للتحقق من صحة البيانات

الكائن	الحدث	الشرح	إلغاء السلوك الافتراضي
أداة التحكم	BeforeUpdate	قبل تحديث بيانات جديدة أو صغيرة في مخزن السجل المؤقت	Yes

الجدول ١٢-١: أحداث شائعة تستخدم للتحقق من صحة البيانات

الكائن	الحدث	الشرح	إلغاء السلوك الافتراضي
	Exit	قبل مغادرة أداة التحكم	Yes
النموذج	BeforeUpdate	قبل تحديث سجل جديد أو متغير في الجدول	Yes
	Delete	قبل حذف السجل	Yes

عندما تستطيع إلغاء السلوك الافتراضي الذي يتبع حدثاً ما، فهذا يعني أن لإجراء الحدث وسيطة Cancel. وعند إعداد وسيطة Cancel لتصبح True في الإجراء فيمكن القول بأن هذا يقوم بإلغاء السلوك الافتراضي الذي يتبع الحدث . أما بالنسب للسلوك الافتراضي الذي يتبع حدث BeforeUpdate الذي تتعرف عليه أداة التحكم فيكون تحديث أكسس للقيمة المتغيرة في مخزن أداة التحكم المؤقت إلى مخزن السجل المؤقت. وعند استخدام أحد أحداث BeforeUpdate لإحداث إجراء التحقق من الصحة، يمكن إلغاء التحديث في حالة عدم اجتياز البيانات لاختبار التحقق من الصحة. ومن ناحية أخرى عند استخدام حدث AfterUpdate لإحداث إجراء التحقق من الصحة يكون أكسس قد قام بالفعل بتحديث المخزن المؤقت بحيث لا تستطيع إلغاء التحديث: "هناك بعض الأحيان يمكنك فيها استخدام حدث AfterUpdate لإحداث إجراء التحقق من الصحة فعلى سبيل المثال في حالة استخدام حدث AfterUpdate للنموذج، قد يمكنك تضمين تعليمات للتراجع عن التحديث وذلك عن طريق حذف السجل المحفوظ".

وكمثال على ذلك، يستخدم إجراء RequiredDate_BeforeUpdate حدث BeforeUpdate لإحداث إجراء التحقق من الصحة لمربع النص RequiredDate. ويحسن هذا الإجراء من كفاءة قاعدة التحقق من الصحة Validation Rule وذلك بإلغاء التغيير حتى لا ينبغي على المستخدم نقر زر Esc.

```
Private Sub RequiredDate_BeforeUpdate(Cancel As Integer)
  If RequiredDate <= OrderDate Then
    MsgBox "The required date must be greater than the order
date."
    Me!RequiredDate.Undo
    Cancel = True
  End If
End Sub
```

إذا كان التاريخ في مربع نص RequiredDate أقل من أو يساوي القيمة المذكورة في مربع نص OrderDate فسيعرض الإجراء رسالة ويستخدم طريقة Undo لأداة تحكم مربع النص لإلغاء التغيير ثم يقوم بإعداد وسيطة Cancel لتصبح True وذلك لإلغاء تحديث مخزن السجل المؤقت وعلى المستخدم تغيير البيانات في أداة التحكم لتصبح تاريخاً صالحاً أو شريط أدوات خارج أداة التحكم دون إحداث أية تغييرات.

لاختبار إجراء الحدث، اختر نموذج Orders في نافذة Database وانقر زر Code في شريط الأدوات. سيتم فتح نموذج Orders في عرض Design وسيتم عرض وحدة النموذج النمطية. ادرج الإجراء RequiredDate_BeforeUpdate ثم انتقل إلى عرض Form. غير RequiredDate ليصبح له قيمة أقل من Order Date ثم اضغط Enter. وسيقوم الإجراء بعرض الرسالة وإلغاء التحديث.

تغيير توقيت اختبار التحقق من الصحة

عند تصميم جدول البيانات عليك بتحديد كيفية إدخال الزر الأساسي، ويمكنك استخدام الطرق الآتية:

- قم بتعيين أرقام متتالية تلقائياً باستخدام حقل AutoNumber على أنه المفتاح الأساسي.
- قم بإنشاء التعبيرات الخاصة بك لتعيين قيم منفردة تلقائياً.
- اسمح بإدخال المفتاح الأساسي كجزء من إدخال البيانات.

ومهما كانت الطريقة التي يتم بها إدخال القيمة فإن أكسس يبحث عن القيم المكررة عند محاولة حفظ السجل. وفي حالات كثيرة يكون انسب وقت لاختبار انفراد المفتاح الأساسي بعد ترك المستخدم لأداة تحكم المفتاح الأساسي مباشرةً بدلاً من الانتظار حتى يقوم المستخدم بإدخال قيم في جميع أدوات تحكم البيانات ثم يحاول بعد ذلك حفظ السجل. يمكن إنشاء إجراء حدث للتعامل مع اختبار الانفراد بنفسك ولتشغيل إجراء الحدث فور محاولة المستخدم تحديث أداة التحكم المتغيرة. أي عندما تتعرف أداة التحكم المتغيرة على حدث BeforeUpdate.

وكمثال على كيفية البحث عن القيم المكررة للمفتاح الأساسي يختبر إجراء CustomerID_BeforeUpdate تفرد القيمة التي يتم إدخالها في أداة التحكم CustomerID في نموذج Customers.

```
Private Sub CustomerID_BeforeUpdate(Cancel As Integer)
    If DCount("*", "Customers", "CustomerID = "_
        & "Screen.ActiveForm.CustomerID") Then
        MsgBox "There is another customer with the same CustomerID."
        Cancel = True
```

يستخدم هذا الإجراء خاصية ActiveForm لكائن Screen للإشارة إلـــى النمـــوذج النشـــط "لاحظ أنه لا يمكنك استخدام خاصية Me في إحدى وسائط وظائف المجال الكلية" في حالة وجود سجل مكرر تقوم وظيفة Dcount بإرجاع القيمة رقماً واحداً بحيث يصبح الشرط True. وفـــي هذه الحالة يقوم الإجراء بعرض رسالة وإلغاء التحديث.

وهناك طريقة أخرى للبحث عن سجل مكرر وتكون باستخدام طريقة Seek ويستخدم إصـــدار إجراء CustomerID_BeforeUpdate التالي طريقة Seek للبحث عن السجل المكـــرر فـــي جدول Customers:

```
Private Sub CustomerID_BeforeUpdate(Cancel As Integer)
   Dim rst As Recordset
   Set rst = CurrentDB.OpenRecordset("Customers", dbOpenTable)
   rst.Index = "PrimaryKey"
   rst.Seek "=", Me!CustomerID
   If Not rst.NoMatch Then
      MsgBox "There is another customer with the same
CustomerID."
      Cancel = True
   End If
End Sub
```

يبدأ هذا الإجراء بفتح مجموعة سجلات من نوع الجدول في جدول Customers "يجـــب أن تكون مجموعة السجلات مجوعة من نوع الجدول حتى تستخدم طريقة Seek"، ويستخدم طريقـــة OpenRecordset لإنشاء كائن Recordset آخر يستخدم في عملية Seek. والـــذي ســـيكون موجوداً مع كائن Recordset الذي يقترن بنموذج Customers. وبعد ذلك يقوم الإجراء بإعداد الفهرس الحالي للمفتاح الأساسي ثم يبحث عن سجل في الجدول باستخدام مفتاح أساسي مطـــابق للقيمة في أداة تحكم Customer ID للنموذج. في حالة إيجاد السجل يكون الخاصيـــة NoMatch القيمة False والتعبير Not rst.NoMatch القيمة True. في هذه الحالة يقوم الإجراء بعـــرض رسالة وبإلغاء السلوك الافتراضي التالي. في حالة عدم إيجاد أي سجل فســـينتهي الإجـــراء دون اتباع أي إجراءات أخرى.

ملاحظـــة لا يمكن استخدام خاصية RecordsetClone للإشـــارة إلـــى مجموعـــة سجلات النموذج عند استخدام طريقة Seek للبحث عن السجل المكـــرر في جدول. فعند استخدام خاصية RecordsetClone للنموذج للإشارة إلـــى مجموعة سجلات النموذج، لا يتم نسخها كجزء من مرجع نسخة مجموعـــة السجلات، ومن ثم تتسبب في محاولة إعداد فهرس حال في نسخة مجموعة السجلات في إحداث خطأ وقت التشغيل

إضافة أزرار أوامر لعمليات إدخال البيانات

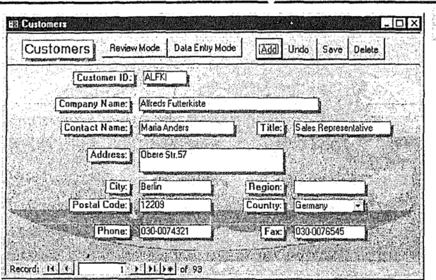
سنستخدم Command Button Wizard لإنشاء مجموعة أزرار أوامر لأتمتة العمليات الأربع الأساسية لإدخال البيانات حتى أن جميع النماذج البسيطة لإدخال البيانات تحتاج إلى إضافة سجل جديد والتراجع عن بعض التغييرات التي أحدثت بالسجل وحفظ التغييرات وحذف السجل.

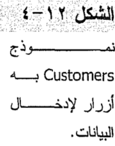
افتح نموذج Customers في عرض View وانقر أداة Control Wizard في مربع الأدوات لتنشيط Control Wizards. انقر أداة Command Button وضع أربعة أزرار أوامر في مقطع المقدمة للنموذج واستخدم فئة Record Operations لـ Command Button Wizard لإنشاء الأزرار كالتالي:

الإجراء	نص الزر خاصية Caption	الاسم
إضافة سجل جديد	Add	CmdAdd
التراجع عن سجل	Undo	CmdUndo
حفظ سجل	Save	CmdSave
حذف سجل	Delete	CmdDelete

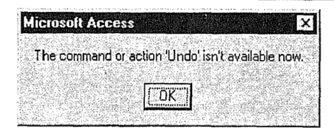
احفظ النموذج ثم انتقل إلى عرض Form "انظر الشكل ١٢-٤" انقر زر Data Entry Mode ثم اختبر الأزرار:

- انقر Add: في أول مرة تقوم فيها بنقر زر Add يعرض أكسس السجل الخالي إذا نقرت مرة أخرى بعض عرض السجل الخالي فلن تكون هناك استجابة.
- انقر Undo إذا ما قمت بتغيير القيمة في إحدى أدوات التحكم ثم نقرت زر Undo سيتم التراجع عن التغيير وإذا قمت بتغيير السجل ثم نقرت زر Save لحفظ كل التغييرات التي أحدثت بالسجل ثم نقرت زر Undo، سيتم التراجع أيضاً عن التغييرات. وإذا لم تحدث أي تغييرات ونقرت زر Undo فسيتم عرض رسالة افتراضية تنم عن أن هناك خطأ ما "انظر الشكل ١٢-٥".



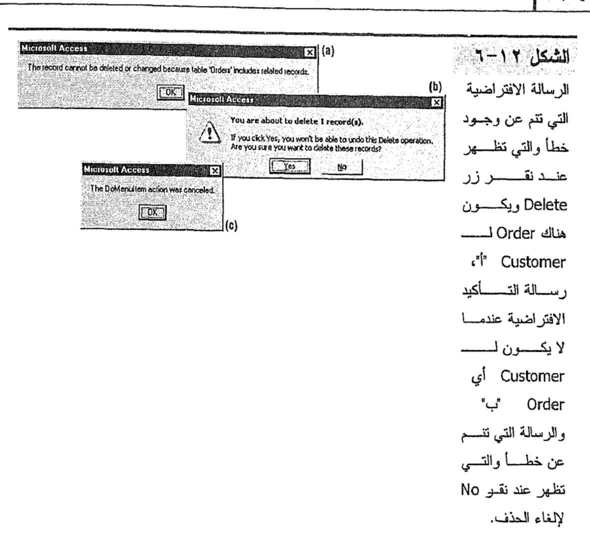


الشكل ١٢-٤ نموذج Customers به أزرار لإدخال البيانات.



الشكل ١٢-٥ الرسالة الافتراضية التي تنم عن وجود خطأ والتي تظهر عند نقر زر Undo في حين أنه لا يوجد تغييرات يتم التراجع عنها.

- انقر زر Save إذا ما قمت بتغيير السجل ثم نقرت زر Save، سيتم حفظ السجل على الرغم من أن التغيير يكون مرضياً لقواعد التحقق من صحة السجل). وإذا لم تقم بتغيير السجل، لن تكون هناك أية استجابة عند نقر زر Save.
- انقر Delete: إذا نقرت زر Delete لنموذج Customer بالترتيبات فستعرض الرسالة الافتراضية التي تنم عن وجود خطأ الموضحة في الشكل ١٢-٦ أ ولن يحذف السجل. وفي العلاقة بين Customers وOrders لا يتم مراجعة "البحث عن" خيار Cascade Delete Related Records فلا تستطيع حذف أي Customer له Order. أما إذا كان Customer لا يمتلك أي Order مثل Customer الذي له CustomerID "FISSA" سجل "٢٢" فستعرض رسالة التأكيد الافتراضية "انظر الشكل ١٢-٦ ب". إذا نقرت على No لإلغاء الحذف سيعرض أكسس الرسالة الافتراضية الموضحة في الشكل ١٢-٦ ج.



الشكل ١٢-٦ الرسالة الافتراضية التي تنم عن وجود خطأ والتي تظهر عند نقر زر Delete ويكون هناك Order لـ Customer "أ"، رسالة التأكيد الافتراضية عندما لا يكون لـ Customer أي Order "ب" والرسالة التي تنم عن خطأ والتي تظهر عند نقر No لإلغاء الحذف.

انتقل إلى عرض Design ونقر زر Code في شريط الأدوات لعرض الإجراءات التي أنشأها Wizard. حدد زر الأمر cmdAdd من قائمة التحرير والسرد Object على الجانب الأيسر أسفل شريط عنوان الوحدة النمطية. والإجراء التالي هو إجراء الحدث الذي أنشأه Command Button Wizard لزر Add:

```
Private Sub cmdAdd_Click()
On Error GoTo Err_cmdAdd_Click
   DoCmd.GoToRecord , , acNewRec
Exit_cmdAdd_Click:
Exit Sub
Err_cmdAdd_Click:
   MsgBox Err.Description
   Resume Exit_cmdAdd_Click
End Sub
```

في كل إجراء من الإجراءات الأربعة يتضمن Command Button Wizard معالج أخطاء عام يعرض رسالة افتراضية تنم عن وجود خطأ ثم ينتهي الإجراء. ولكل زر يقوم Wizard بتشغيل إحدى طرق كائن DOCmd ولا تفشل الإجراءات التي تستخدم الأزرار لإضافة سجل جديد أو حفظ سجل موجود بالفعل.

لإضافة سجل جديد يقوم wizard بتشغيل طريقة GoToRecord. بعدما يتم عرض السجل الحالي يختفي أمر New Record من القائمة الثانوية لأمر GoTo في قائمة Edit ولا تفشل طريقة GoToRecord طالما أن السجل الجديد معروض بالفعل.

لحفظ التغييرات يستخدم wizard طريقة DoMenuItem لتشغيل أمر Save Record في قائمة Reccords ويكون هذا الأمر متوفراً بالنسبة للسجلات الموجودة بالفعل وبالنسبة للسجل الخالي سواء قمت أم لم تقم بأية تغييرات في السجل. في حالة عمل تغييرات تكون غير كافية بالنسبة لقواعد التحقق من صحة السجل، يعرض أكسس الرسالة الافتراضية أو يعرض رسالة التحقق من الصحة مخصصة ولا يقوم بتحديث السجل. أما بالنسبة لطريقة VBA فهي تتجاهل هذا السلوك.

قد تفشل أحياناً الإجراءات التي تستخدم الأزرار للتراجع عن التغييرات أو لحذف سجل وفي هذه الحالة يتم عرض رسالة تنم عن أن هناك خطأ ما وذلك لأن VBA لا يستطيع تشغيل الطريقة. ولنستكشف معاً التعليمات البرمجية والتعديلات التي يمكن إحداثها لتجنب ظهور مثل هذا النوع من الرسائل.

تعديل الإجراء للتراجع عن تغيير

للتراجع عن تغييرات بالسجل يستخدم wizard طريقة DoMenuItem لتشغيل أمر Undo المضمن. عند العمل مع بطريقة تفاعلية "باستخدام القوائم" مع عدم إحداث أي تغييرات بالسجل، يتحول لون ثمة بطريقة ستخدام زر الأمر) مع عدم إحداث أي تغييرات بالسجل، تفشل الطريقة وذلك لأن الأمر لا يكون موجوداً. تظهر أوامر Undo فقط عندما تكون قد قمت بالفعل بإحداث تغير ما السجل ويمكن استخدام خاصية النموذج Dirty لاختار لي قد تم تغييره قبل إصدار الأمر. ويكون لخاصية Dirty القيمة True طالما أن السجل الحالي قد تم تعديله بعد آخر مرة تم فيها حفظه، فيما عدا ذلك تكون القيمة False.

وبالتعديل التالي لن تحصل على أية استجابة عند نقر زر Undo لسجل لم يتم تغييره بالإضافة إلى ذلك عند تغيير أو حفظ سجل، لن تستطيع استخدام زر Undo ثانية للتراجع عن التغييرات التي أحدثتها آخر سجل تم حفظه. "بعد حفظ السجل تظل قيمة خاصية Dirty للنموذج False إلى أن تحدث أية تغييرات أخرى بالسجل".

```
Private Sub cmdUndo_Click()
   On Error GoTo Err_cmdNew_Click
   If Me.Dirty Then
      DoCmd.DoMenuItem acFormBar, acEditMenu, acUndo, ,
acMenuVer70
End If
Exit_cmdNew_Click:
Exit Sub
Err_cmdNew_Click:
   MsgBox Err.Description
   Resume Exit_cmdNew_Click
End Sub
```

تعديل الإجراء لحذف السجل

لحذف سجل يقوم wizard إنشاء الإجراء التالي:

```
Private Sub cmdDelete_Click()
   On Error GoTo Err_cmdDelete_Click

      DoCmd.DoMenuItem acFormBar, acEditMenu, 8, , acMenuVer70
      DoCmd.DoMenuItem acFormBar, acEditMenu, 6, , acMenuVer70
Exit_cmdDelete_Click:
Exit Sub
Err_cmdDelete_Click:
   MsgBox Err.Description
   Resume Exit_cmdDelete_Click
End Sub
```

إذا كان حذف السجل المحدد سيتسبب في انتهاك القواعد التمامية المرجعية، وفي تتالي الخيارات التي يتم البحث عنها من أجل العلاقات بين جدول البيانات والجداول الأخرى في قاعدة البيانات، فسيقوم أكسس بعرض رسالة التأكيد الافتراضية "انظر الشكل ١٢-٦ ج" وذلك لأن نقر زر No يمنع VBA من تنفيذ أمر حذف السجل. ويمكن تعديل إجراء cmdDelete_Click لكبح رسالة الخطأ كالتالي:

فقط حدد تعليمات الخطأ البرمجية لرسالة الخطأ الافتراضية الموضحة في الشكل ١٢-٦ج وذلك بتغيير التعليمات البرمجية لتناول الخطأ الخاصة الإجراء لتصبح كالتالي:

```
Err_cmdDelete_Click:
   MsgBox Err.Number & Err.Description
Resume Exit_cmdDelete_Click
```

اعرض Customer باستخدام CustomerID FISSA. انقر زر Delete وانقر No لإلغاء الحذف. سيعرض مربع الرسالة تعليمات الخطأ البرمجية ٢٥٠١. ثم انقر Ok للإغلاق مربع الرسالة.

قم بتعديل التعليمات البرمجية التي تتناول الخطأ لاختبار تعليمات الخطأ البرمجية ثم أنه الإجراء بدون عرض الرسالة إذا كانت تعليمات الخطأ البرمجية تساوي ٢٥٠١ كما هو موضح بأسفل. وتتناول جمل Else الأخطاء غير المتوقعة:

```
Private Sub cmdDelete_Click()
   On Error GoTo Err_cmdDelete_Click
      DoCmd.DoMenuItem acFormBar, acEditMenu, 8, , acMenuVer70
      DoCmd.DoMenuItem acFormBar, acEditMenu, 6, , acMenuVer70
Exit_cmdDelete_Click:
Exit Sub
Err_cmdDelete_Click:
   If Err.Number = 2501 Then
      Exit Sub
     Else
       MsgBox Err.Description
       Resume Exit_cmdDelete_Click
     End If
  End Sub
```

احفظ الوحدة النمطية واختبر الإجراء المعدل عن طريق عرض customer باستخدام CustomerID FISSA ثم انقر زر Delete ثم No.

حذف السجل عندما يكون له سجلات أخرى متعلقة به

عند محاولة حذف سجل من جدول باستخدام نموذج له خاصية AllowDeletions معدة لتكون Yesستعتمد الاستجابة على إذا ما كان الجدول يتعلق بجداول أخرى في علاقة تؤكد فيها التمامية المرجعية، وعلى ما إذا كان السجل الذي تريد حذفه ستعلق بسجلات أخرى في جدول آخر. فعلى سبيل المثال يتعلق الجدولان Customers وorders ببعضهما البعض، ويتم البحث عن خيار Enforce Referential Integrity. وعندما لا يكون لـ Customers معين أي أوامر

orders يمكن حذف السجل الخاص، أما إذا كان له orders فستعتمد الاستجابة لذلك على إذا كان قد تم البحث عن خيار Cascade Delete Related records.



يعد Cascade Update Related Fields وcascade Delete Related Records خياران يمكن إعدادهما عند إنشاء علاقة بين جدولين وبالافتراض يتم تحديد Cascade Update Related Fields، مما يعني أن أي تغييرات تحدثها في أحد الجداول ستنتقل إلى الجداول الأخرى التي تتعلق بهذا الجدول وبالافتراض أيضاً لا يتم تحديد خيار Cascade Delete Related Records. إذا رغبت في تغيير واحداً من هذين الإعدادين عليك أولاً إغلاق جميع الجداول ثم تحديد Tools ⇦ Relationships وفي نافذة Relationships انقر الخط الذي يصل بين الجدولين نقراً مزدوجاً والذي يمثل أيضاً العلاقة بين هذه الجداول. يمكنك بعد ذلك إحداث التغييرات اللازمة Cascade Referential Integrity.

عند تحديد خيار Cascade Delete Related Records يسمح أكسس بحذف Customers وOrders المتعلقة به في نفس الوقت، أما إذا لم يتم تحديد هذا الخيار فلن يسمح أكسس بحذف Customers له Orders متعلقة به. في هذه الحالة يمكن استخدام إجراء حدث كالإجراء التالي. "يمكن أيضاً استخدام هذا الإجراء إذا لم ترغب في السماح بحذف سجل له أخرى متعلقة به، بغض النظر عن خيار التتالي Cascade".

```
Private Sub cmdDelete_Click()
    Dim intNumber As Integer
    On Error GoTo Err_cmdDelete_Click
    intNumber = DCount("*", "Orders", "CustomerID = " _
       & "Screen.ActiveForm.CustomerID")
    If intNumber = 0 Then
       RunCommand acCmdDeleteRecord
    Else
       MsgBox ("There are " & intNumber & _
          " orders so you can't delete this customer.")
    End If
Exit_cmdDelete_Click:
Exit Sub
```

```
Err_cmdDelete_Click:
   If Err.Number = 2501 or Err.Number = 2046 Then
      Exit Sub
   Else
      MsgBox Err.Description
      Resume Exit_cmdDelete_Click
   End If
End Sub
```

يستخدم هذا الإجراء وظيفة Dcount لتحديد عدد السجلات المتعلقة بالسجل المعروض في النموذج. في هذا المثال تحدد وظيفة Dcount عدد الترتيبات Orders الخاصة بالعميل الحالي. إذا لن يكن للعميل Customer أي ترتيبات Orders سيسمح الإجراء بالحذف، أما إذا كان للعميل ترتيبات، فسيعرض الإجراء رسالة بعدد الترتيبات Orders ثم ينتهي.

تتالي الحذف

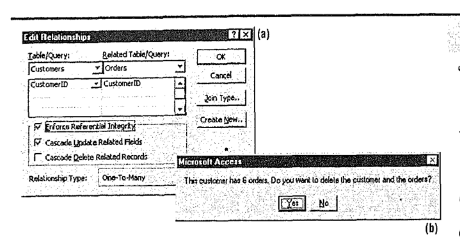
عندما يكون للسجل سجلات أخرى متعلقة به، يتم اختيار خيار التتالي Cascade وإذا أردت السماح بحذف السجل المعروض وكل السجلات الأخرى المتعلقة به، يمكنك استخدام إجراء حدث كالإجراء التالي:

```
Private Sub cmdDelete_Click()
   Dim intNumber As Integer
   On Error GoTo Err_cmdDelete_Click
   intNumber = DCount("*","Orders", "CustomerID = " _
      & "Screen.ActiveForm.CustomerID")
   If intNumber = 0 Then
       DoCmd.DoMenuItem acFormBar, acEditMenu, 7, , acMenuVer70
   Else
      If MsgBox ("This customer has " & intNumber & _
         " orders. Do you want to delete the customer " _
         & "and the orders?", vbYesNo) = vbYes Then
         RunCommand acCmdDeleteRecord
      End If
   End If
Exit_cmdDelete_Click:
```

```
Exit Sub
Err_cmdDelete_Click:
   If Err.Number = 2501 Then
      Exit Sub
   Else
      MsgBox Err.Description
      Resume Exit_cmdDelete_Click
   End If
End Sub
```

يبدأ هذا الإجراء أيضاً بتحديد عدد السجلات المتعلقة بالسجل ثم السماح بالحذف إذا لم يكن ذلك متوفراً بالفعل. عندما يكون للسجل سجلات أخرى متعلقة به، فإن الإجراء يستخدم وظيفة MsgBox كشرط لجملة If. وتعرض وظيفة MsgBox رسالة بعدد السجلات المتعلقة بالسجل الأساسي، وتطلب هذه الرسالة أيضاً من المستخدم اتخاذ قرار بشأن حذف السجل والسجلات الأخرى المتعلقة به. ويتوقف تنفيذ الإجراء إلى أن ينقر المستخدم أحد الأزرار بمربع الرسالة، عندئذ يستطيع أكسس تقييم وظيفة MsgBox واستكمال تنفيذ الإجراء باستخدام القيمة التي ترجعها وظيفة MsgBox. وعند نقر زر Yes يحذف السجل وجميع السجلات الأخرى المتعلقة به، فيما عدا ذلك ينتهي الإجراء.

لاختبار هذا الإجراء اغلق نموذج Customer "إذا كان مفتوحاً" واختر Tools ⇦ Relationships انقر بعد ذلك خط العلاقة بين جدولي Customers وOrders نقراً مزدوجاً لعرض مربع Edit Relationships انظر الشكل "١٢-٧ أ" واختر خيار Cascade Delete Related Records وانقر Ok ثم اغلق النافذة Relationships. بعد ذلك حدد نموذج Customers في نافذة Database وانقر زر Code في شريط الأدوات. عدل إجراء الحدث الموضح سلفاً، وأخيراً احفظ الوحدة النمطية. وبعد عرض العميل "Customer" الأول في نموذج Customers انقر زر Delete، وسيقدم مربع الرسالة اختياراً. "انظر الشكل ١٢-٧ ب".



الشكل ١٢-٧ اختر خيارات التتالي "Cascade" في مربع حوار Edit Relationships "أ". سيعرض الإجراء المراجع مربع حوار يوفر للمستخدم اختياراً بشأن حذف العميل Customer والترتيبات (Orders).

ترحيل "نقل" القيم إلى سجل جديد

غالباً ما تكون بيانات الحقول المختلفة في السجل الجديد مطابقة للبيانات الموجودة بسجل آخر. وبدون الأتمتة يكون الحل الوحيد هو إعادة إدخال القيم المكررة يدوياً في السجل الجديد غير أنه يمكن تطوير تقنيات البرمجة لتعبئة القيم المتكررة مما يقلل بالتالي من العمل اليدوي.

ويشرح هذا المقطع تقنيتان لنقل القيم من سجل إلى آخر، فالتقنية الأولى مصممة لأن تنقل القيم من السجل الحالي إلى السجل التالي الجديد، كما أنها تقوم بإعداد خاصية DefaultValue لأدوات التحكم التي تريد نقل قيمها إلى السجل التالي الجديد. أما التقنية الثانية، فهي مصممة لأن تنقل القيم من أي سجل إلى "ليس بالضرورة السجل الحالي" إلى السجل التالي الجديد. غير أنك في كلتا التقنيتين تقوم بنقل بعض القيم، كما أن كلتا التقنيتين تستخدمان خاصية أداة التحكم Tag لتحديد ما إذا كانت قيمة أداة التحكم سيتم نقلها.

إعداد خاصية القيمة الافتراضية DefaultValue

عند إعداد خاصية DefaultValue لحقل في جدول أو عرض Design أو لأداة تحكم في نموذج عرض Design يقوم أكسس تلقائياً بإدخال قيمة خاصية DefaultValue في أداة التحكم وذلك عند عرض سجل جديد في النموذج. ويعد إعداد خاصية DefaultValue في عرض Design هاماً

عند استخدام نفس التعبير لحساب القيمة التي سيتم إدراجها في أداة التحكم لكل سجل جديد. فعلى سبيل المثال تعد خاصية DefaultValue لأداة التحكم OrderDate في نموذج Orders لتكون Date()= حتى يعرض كل ترتيب جديد التاريخ الحالي تلقائياً.

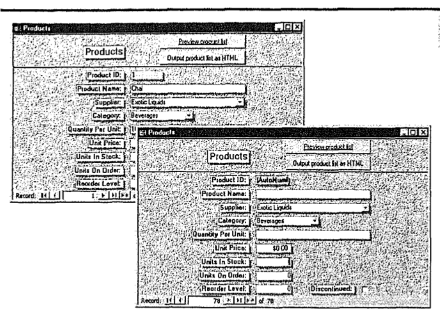
كما يمكن إعداد خاصية DefaultValue بطريقة مبرمجة، فقد تريد مثلاً نقل القيم من سجل إدخال البيانات الحالي إلى سجل جديد. وغالباً ما يكون نقل القيم من سجل إلى آخر طريقة عالية الكفاءة لدفع إدخال البيانات عندما يتغير حقل أو أكثر بغير مواظبة.

وكمثال على ذلك، افترض مثلاً أنك قد بدأت مؤخراً العمل مع مورد جديد يقدم العديد من المنتجات في فئة واحدة أو فئتان، وتريد إضافة المنتجات الجديدة. في هذه الحالة تكون الوسيلة المناسبة لتحديث البيانات هي نقل القيم إلى أدوات التحكم SupplierID وCategoryID في نموذج Products. فبعد حفظ بيانات منتج جديد أو منتج متغير وبعد نقر زر New "أحد أزرار التنقل الافتراضية بالنموذج" للانتقال إلى سجل جديد خالي، ستظهر القيم الموجودة في أداتين التحكم هاتين تلقائياً في السجل الجديد. ويقوم إجراء الحدث Form_AfterUpdate التالي بأتمتة العملية:

```
Private Sub Form_AfterUpdate()
   Dim ctl As Control
   For Each ctl In Screen.ActiveForm.Controls
      If ctl.Tag = "Carry" Then
         ctl.DefaultValue = """" & ctl.Value & """"
      End If
   Next
End Sub
```

لكلتا أداتي التحكم قيمة نص بحيث يصبح من المفترض أن يقوم الإجراء بتقييمها. وتكون النتيجة سلسلة (...) داخل سلسلة أخرى، ومن ثم يجب استخدام إحدى مجموعات الرموز التي يتعرف عليها أكسس للحد من وجود سلسلة داخل سلسلة أخرى. يستخدم هذا الإجراء أزواجاً من علامات التنصيص المزدوجة لتحديد السلسلة الداخلية (...). وأخيراً تحيط السلسلة بعلامات تنصيص مزدوجة لنوضح أن النتيجة النهائية سلسلة (...).

لاختبار هذا الإجراء افتح النموذج Products في عرض Design. حدد أدوات التحكم SupplierID وCategoryID واكتب Carry في خاصية Tag الخاصة بهم "يمكن استخدام خاصية Tag لتخزين أية معلومات، في هذه الحالة تستخدم خاصية Tag لتعليم أدوات التحكم المراد نقل قيمها إلى سجل آخر" ثم انقر زر Code في شريط الأدوات لعرض وحدة النموذج النمطية. إنتقل بعد ذلك إلى عرض Form وغير أي أداة تحكم في السجل الحالي، ثم الوحدات المخزنة ثم انقر زر New، وسيتم نقل قيم المورد والفئة إلى سجل آخر. "انظر الشكل ١٢-٨".



الشكل ١٢-٨
يمكن استخدام خاصية Tag لتعليم أداة التحكم التي ستنتقل قيمتها إلى سجل آخر، ثم استخدام إجراء لإعداد خاصية DefaultValue للقيمة المعنية لأداة التحكم.

استخدام قيم افتراضية مخصصة

طريقة أخرى لإدخال بيانات أسرع تتضمن البحث عن معلومات في سجل آخر ونقل البيانات من هذا السجل إلى السجل الجديد. افترض مثلاً أن العملاء الذين يضعون ترتيبات جديدة بطريقة مطابقة لهم نفس الترتيب الذي يشحن على عنوان لا يتغير كثيراً.

في Northwind يصمم نموذج Orders بحيث أنه عند تحديد العميل من مربع السرد والتحرير، يتم اعتبار اسم وعنوان الشحن افتراضياً نفس اسم وعنوان العميل. في هذه الحالة يقوم إجراء الحدث CustomerID_AfterUpdate بنسخ اسم وعنوان العميل في أدوات تحكم الشحن المناسبة. وكمثال على نسخ المعلومات من سجل إلى آخر، سنقوم بتغيير إجراء الحدث، بحيث بدلاً من نسخ اسم وعنوان العميل يقوم VBA بالبحث عن آخر ترتيب وضعه العميل، ثم ينسخ معلومات عن الشحن من هذا الترتيب لتصبح هي ذاتها معلومات الشحن في الترتيب الثاني، ويتحقق ذلك من خلال الإجراء التالي:

```
Private Sub CustomerID_AfterUpdate()
    Dim rst As RecordSet
    Dim strCriteria As String
    strCriteria = "CustomerID = """ & CustomerID & """"
    Set rst = Me.RecordsetClone
    rst.FindLast strCriteria
    If Not rst.NoMatch Then
        ShipName = rst!ShipName
```

```
        ShipAddress = rst!ShipAddress
        ShipCity = rst!ShipCity
        ShipRegion = rst!ShipRegion
        ShipPostalCode = rst!ShipPostalCode
        ShipCountry = rst!ShipCountry
    End If
End Sub
```

تجد طريقة FindLast للعميل آخر ترتيب تم تحديده في مربع التحرير والسرد CustomerID الذي يحتوي على قيمة للبحث. عند إنشاء معيار السلسلة لطريقة FindLast فإنك ستحتاج إلى إرغام VBA على توضيح القيمة المحددة في مربع التحرير والسرد قبل أن يستخدم الإجراء معيار البحث لإيجاد السجل، أي يمكن القول أن معيار السلسلة لطريقة FindLast يحتوي على سلسلة داخلية. في هذا المثال التالي ستجد أن CustomerID له قيمة نص، بحيث يمكن استخدام أياً من التعبيرات الثلاثة التالية للمعيار وChr$(34) هو تمثيل ANSI لأزواج من علامات التنصيص المزدوجة.

```
strCriteria = "CustomerID = """ & CustomerID & """"
strCriteria = "CustomerID = '" & CustomerID & "'"
strCriteria = "CustomerID = " & Chr$(34) & CustomerID & Chr$(34)
```

لكل قيمة تنقل إلى سجل آخر، يستخدم الإجراء جملة تعيين لتعيين قيمة أداة التحكم بالنموذج كقيمة الحقل الذي يحمل نفس الاسم في سجل الترتيب الأخير أي أن كل تعيين يطابق أداة تحكم ما في مجموعة النموذج Controls بحقل ما في تعيين تطابق أداة تحكم ما في مجموعة السجلات Fields. ويكون هذا التطابق نتيجة لأن نموذج Orders يكون مصمماً بحيث يكون لكل أداة تحكم منضمة بالنموذج نفس اسم الحقل الذي تنضم إليه.

لاختبار هذا الإجراء افتح نموذج Orders في عرض Design، وحدد أدوات التحكم ShipName وShipAddress وShipCity وShipRegion وShipPostalCode وshipCountry. انقر بعد ذلك خاصية Tag في ورقة Multiple لتحديد الخاصية ثم اكتب Carry، بحيث تصبح أدوات التحكم المراد نسخها معلمة. انقر الزر Code في شريط الأدوات لعرض وحدة النموذج النمطية ثم عدل عبارات إجراء الحدث CustomerID_AfterUpdate كما هو موضح سلفاً.

سينجح هذا الإجراء لكنه لا يعد إجراءاً رائعاً حيث أن لكل أداة تحكم عبارة تعيين منفردة، بينما الأجراء الرائع هو الذي يقوم بتنفيذ حلقة في كل أدوات تحكم النموذج وفي كل خطوة من خطوات التنفيذ يحدد الإجراء ما إذا كانت قيمة أداة التحكم ستنقل إلى سجل آخر، كما يقوم أيضاً بإعداد قيمة أداة التحكم في قيمة الحقل المناسبة.

للوصول إلى أفضل النتائج، سنقوم بتعديل الإجراء بحيث يتم مطابقة أدوات التحكم المنضمة والحقول تلقائياً عندما تكون الأسماء واحدة. وكل عبارة تعيين تعين قيمة أداة التحكم بقيمة الحقل الذي يحمل الاسم المناسب. وكخطوة أولى سنعيد صياغة عبارة التعيين ونضع لها بناءاً مناسباً لتنفيذ حلقة في مجموعة، فعبارتا التعيين التاليتان مثلاً متطابقتان:

```
ShipAddress = rst!ShipAddress
ShipAddress.Value = rst("ShipAddress")
```

لكن في العبارة الثانية، يذكر الجانب الأيسر بوضوح أن قيمة Value يتم إعدادها يستخدم الجانب الأيمن مرجع لحقل ShipAddress. عند استخدام مرجع الأقواس لحقل يمكنك استبدال السلسلة الحرفية لاسم الحقل بمتغير له نفس القيمة. فعند إنشاء العبارات لتنفيذ حلقة في أدوات التحكم ستحتاج إلى استبدال السلسلة الحرفية مثل ShipAddress بمتغير يحتوي على عداد الحلقة.

أما الجزء التالي من الشرح فهو أكثر تجريداً ويحتاج إلى بعض التعقل عند إنشاء الحلقات. سنقوم بإنشاء حلقة For...Next لتنفيذ حلقة في عناصر مجموعة Controls، لذا ستحتاج إلى استخدام فهرس مرقم للإشارة إلى أدوات تحكم النموذج. فإذا كانت frm تشير إلى النموذج، فسيكون رقم أدوات التحكم بالنموذج frm.Count – 1 ويكون لمجموعة Controls أرقام فهرس تتراوح بين صفر و frm.Count – 1 إذا كان K رقم الفهرس في هذا المعدل المحدد ، إذن:

- frm(k) تشير إلى أداة تحكم ما بالنموذج.
- frm(k).Value هي قيمة أداة التحكم.
- frm(k).Name هو اسم أداة التحكم.

تطابق قيمة أحد أرقام الفهرس أداة التحكم ShipAddress مثلاً بمعنى آخر هناك رقم j ومن ثم:

- frm(j) يشير إلى أداة التحكم ShipAddress.
- frm(j).Value هو قائمة أداة التحكم ShipAddress.
- frm(j).Name هو اسم أداة التحكم "وأيضاً اسم الحقل المنضمة إليه أداة التحكم".

كما يمكن كتابة عبارة التعيين التالية لأداة التحكم ShipAddress لرقم فهرس معين يطابق ShipAddress:

```
frm(j).Value = rst(frm(j).Name)
```

تطابق كل أداة تحكم قيمة مختلفة لرقم الفهرس، وبما أن أرقام الفهرس تتراوح بين صفر وعدد أدوات التحكم الموجودة بالنموذج، فإن كل رقم فهرس يطابق أداة تحكم مختلفة. وعند تنفيذ

حلقة في أدوات تحكم النموذج، فإننا نقوم أولاً باختبار خاصية أداة التحكم Tag فإذا كان لها القيمة Carry، سيقوم الإجراء بالانتقال إلى أداة التحكم التالية بدون إعداد قيمة. ويتضمن الإجراء التالي الحلقة التي تقوم بالمطابقة المؤتمتة:

```
Private Sub CustomerID_AfterUpdate()
    Dim frm As Form
    Dim rst As RecordSet
    Dim strCriteria As String, k as Integer
    strCriteria = "CustomerID = "''" & CustomerID & "''"
    Set frm = Me
    Set rst = frm.RecordsetClone
    rst.FindLast strCriteria
    If Not rst.NoMatch Then
        For k = 0 To frm.Count - 1
            If frm(k).Tag = "Carry" Then
                frm(k).Value = rst(frm(k).Name)
            End If
        Next
    End If
End Sub
```

لاختبار هذا الإجراء عدل العبارات في إجراء CustomerID_AfterUpdate كما هو موضح سلفاً واحفظ الوحدة النمطية. افتح بعد ذلك سجلاً جديداً وحدد Around the Horn في حقل Bill To. سيعرض الإجراء اسم وعنوان شحنة العميل. قم بتغيير اسم الشحنة إلى Butterfield's Bicycles وعنوانها إلى 2 Southdown Lane انقر زر New وحدد Around the Horn في Bill وانقلها إلى مربع التحرير والسرد. حينئذ سيعرض السجل الجديد المعلومات عن الشحنة من آخر ترتيب لهذا العميل.

العمل مع البيانات في نماذج متعلقة ببعضها

قد تحتاج أحياناً إلى فتح نموذجين متعلقين ببعضهما البعض في نفس الوقت، فعلى سبيل المثال، في أثناء مراجعة الترتيبات الخاصة بعميل موجود بالفعل باستخدام نموذج Customer Orders قد تحتاج إلى فتح نموذج Customers لتحرير المعلومات. ومع وجود نموذجين مفتوحين في وقت واحد، ستحتاج إلى التأكد من أن النموذجين يعرضان البيانات الأكثر حداثة. وعند تحرير

سجل موجود بالفعل في نموذج ما، يقوم أكسس تلقائياً بتحديث البيانات المعروضة في النماذج المتعلقة ببعضها بمجرد أن تصبح النموذج النشط. غير أنه عند إضافة سجل جديد أو حذف سجل موجود بالفعل، لا يقوم أكسس تلقائياً بتحديث تلك التغيرات.

أتمتة نموذج Customer Orders

قبل استكشاف التحديثات التلقائية، سنحتاج أولاً إلى أتمتة نموذج Customer Orders. لذا سنضع مربع التحرير والسرد للبحث في نموذج Customer Orders لتحديد وعرض عميل موجود بالفعل في هذا النموذج كما سنضيف زر أمر لفتح وتزامن نموذج آخر وهو نموذج Customers لعرض سجل العميل المحدد وسيستخدم هذا الزر إجراء الحدث التالي:

```
Private Sub cmdCustomer_Click()
    Dim strForm As String, strWhere As String
    strForm = "Customers"
    strWhere = "CustomerID = """ & Me!CustomerID & """"
    DoCmd.OpenForm formname:= strForm, wherecondition:=strWhere
End Sub
```

يمكن استخدام أي من الوسائط التالية في Wherecondition:

```
strWhere = "CustomerID = Forms![Customer Orders]!CustomerID"
strWhere = "CustomerID = Screen.ActiveForm.CustomerID"
strWhere = "CustomerID = """ & Me!CustomerID & """"
strWhere = "CustomerID = """ & Form.CustomerID & """"
```

في كل من التعبيرين الأخيرين، يجب أن تضع متغير السلسلة متسلسلاً بحيث ينتج عنه سلسلة مثل & Me!CustomerID &. ثم استخدم أحد المحددات، كأن تستخدم مثلاً زوجاً من علامات التنصيص المزدوجة، وذلك لتعليم بداية ونهاية السلسلة في إطار تعبير السلسلة الخارجي.

لأتمتة نموذج اتبع الخطوات التالية:

٤- افتح نموذج Customer Orders في عرض Design وقم بإعداد خاصية AllowEdits بحيث تكون Yes. وتقوم بإعداد الخاصية على هذا النحو حتى تستطيع تغيير القيم في مرع التحرير والسرد للبحث.

٥- اختر View ⇦ Footer/Form Header وسنضع كل من زر الأمر ومربع التحرير والسرد للبحث في رأس النموذج.

٦- الغ تنشيط أداة Control Wizard. وفي رأس النموذج، ضع زر أمر يحمل اسم cmdCustomer. له خاصية Caption معدة بحيث تكون Customer Info. ادرج إجراء الحدث cmdCustomer_Click الموضح سلفاً في خاصية الحدث OnClick للزر.

٧- ضع مربع التحرير والسرد للبحث في رأس النموذج، وقم بإعداد خاصية Caption للتعليمية لتكون Lookup Customer ثم قم بإعداد خواص مربع التحرير والسرد كالآتي:

Name	CboFind
RowSourceType	Table/Query
RowSource	Customers
ColumnCount	2
ColumnWidths	0;1.5
BoundColumn	1
LimitToList	Yes
List Width	1.5

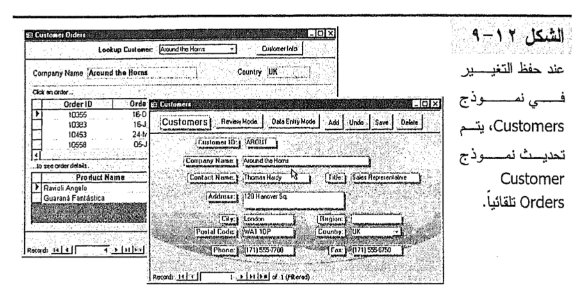
٨- ادخل إجراء الحدث التالي في حدث AfterUpdate لمربع التحرير والسرد:

```
Private Sub cboFind_AfterUpdate()
   Dim strCriteria As String
   strCriteria = "CustomerID = """ & Me!cboFind & """"
   Me.RecordsetClone.FindFirst strCriteria
   Me.Bookmark = Me.RecordsetClone.Bookmark
End Sub
```

٩- احفظ النموذج وانتقل إلى عرض Form. حدد عميلاً ما باستخدام مربع التحرير والسرد للبحث وانقر زر Customer Info.

١٠- انقر زر Data Entry Mode لتغيير وضع نموذج Customers غير الحروف الهجائية لاسم الشركة وانقر زر Save أو اضغط Shift+Enter لحفظ التغيير. لاحظ أن الهجاء المتغير سيعرض تلقائياً في نموذج Customer Orders. "انظر الشكل ١٢-٩".

١١- اغلق نموذج Customers.



الشكل ١٢-٩

عند حفظ التغيير في نموذج Customers، يتم تحديث نموذج Customer Orders تلقائياً.

إضافة صف جديد إلى قائمة التحرير والسرد

عند كتابة اسماً لعميل جديد في مربع التحرير والسرد، يقوم أكسس بعرض العبارة المعهودة التي تنم عن وجود خطأ ما، موضحاً أنه يجب إدخال قيمة في قائمة التحرير والسرد. سنقوم بتعديل نموذج Customer Orders للسماح للمستخدم بإدخال اسم العميل الجديد في مربع التحرير والسرد. وعندما يقارن أكسس النص الذي تم إدخاله بالأسماء الموجودة في قائمة التحرير والسرد ويجد أن هناك اسماً ما ليس موجوداً بالقائمة، في هذه الحالة يتعرف مربع التحرير والسرد على حدث NotInList. وسنقوم بإنشاء إجراء حدث لحدث NotInList.

ويكون بناء جملة تعريف إجراء حدث NotInList كالآتي:

```
Private Sub controlname_AfterUpdate(NewData As String, Response As
Integer)
```

حيث:

- **controlname:** اسم أداة التحكم.
- **Newdata:** هي الوسيطة السلسلة التي تحمل النص الذي كتبه المستخدم في مربع التحرير والسرد. يستخدم أكسس وسيطة NewData لتمرير النص إلى إجراء الحدث.
- **Response:** الثابت الحقيقي الذي تقوم بإعداده في الإجراء لتخبر أكسس بكيفية الرد على الحدث. يمكن إعداد Response لأي من الثوابت الثلاثة التالية:

AcDataErrDisplay: يعرض الرسالة الافتراضية التي تنم عن وجود خطأ ما. استخدم هذا الثابت عندما ترفض السماح للمستخدم بإضافة قيمة جديدة لقائمة التحرير والسرد.

AcDataErrContinue: لا يعرض الرسالة الافتراضية التي تنم عن وجود خطأ ما. استخدم هذا الثابت لمنع أكسس من عرض رسالة الخطأ الافتراضية.

AcDataErrAdded: لا يعرض الرسالة الافتراضية التي تنم نع وجود خطأ ما، لكن يسمح لك بإضافة إدخال في قائمة مربع التحرير والسرد. استخدم هذا الثابت عندما يتضمن إجراء الحدث عبارات تضيف القيمة إلى حقل في مصدر البيانات المتضمن في مربع التحرير والسرد. ويعتمد تصميم الإجراء على خواص RowSourceType وRowSource لمربع التحرير والسرد. فبعد إضافة الإدخال إلى القائمة يطلب أكسس تلقائياً مربع التحرير والسرد.

يسمح لك إجراء الحدث التالي بإضافة ما تشاء في مربع التحرير والسرد:

```
Private Sub CustomerID_NotInList(NewData As String, _
    Response As Integer)
  Dim intNew As Integer
  intNew = MsgBox ("Do you want to add a new customer?", vbYesNo)
  If intNew = vbYes Then
    RunCommand acCmdUndo
    Response = acDataErrContinue
    DoCmd.OpenForm formname:="Customers",datamode:=
acFormAdd
    Forms!Customers!CompanyName = NewData
  Else
    MsgBox "The company name you entered isn't an existing " _
      & "customer."
    RunCommand acCmdUndo
    Response = acDataErrContinue
  End If
End Sub
```

يبدأ هذا الإجراء بعرض مربع رسالة يسأل ما إذا كنت تريد إدخال عميل جديد. في حالة نقر زر Yes، يتراجع الإجراء عن الإدخال في مربع التحرير والسرد، ويوقف رسالة الخطأ الافتراضية، ثم يفتح النموذج Customers الذي يكون جاهزاً لإدخال البيانات في سجل خال وينقل الاسم الذي كتبته كالاسم الجديد للشركة. أما في حالة نقر زر No يعرض الإجراء رسالة ثم يتراجع عن الإدخال في مربع التحرير والسرد.

لاختبار الإجراء، انتقل من نموذج Customer Orders إلى عرض Design انقر بالماوس في مربع التحرير والسرد Lookup Customer وانقر خاصية الحدث On Not In List ثم انقر

زر Build على يمين مربع الخواص. بعد ذلك قم بإنشاء إجراء الحدث المذكور سلفاً ثم احفظ النموذج. انتقل بعد ذلك إلى عرض Form واكتب Denver Delights كاسم لعميل جديد. في قائمة التحرير والسرد، واضغط Enter ثم انقر زر Yes في مربع الرسالة حينئذ سيعرض أكسس سجلاً جديداً لإدخال البيانات نه كاسم الشركة في نموذج customers.

انقر بعد ذلك زر Data Entry لتغيير النموذج إلى وضع إدخال البيانات، وادخل DENVE على أنه CustomerID. ثم اختر USA في مربع التحرير والسرد Country وادخل أي معلومات أخرى عن العميل الجديد. ثم اغلق نموذج Customers وانقر السهم إلى أسفل في مربع التحرير والسرد Lookup.

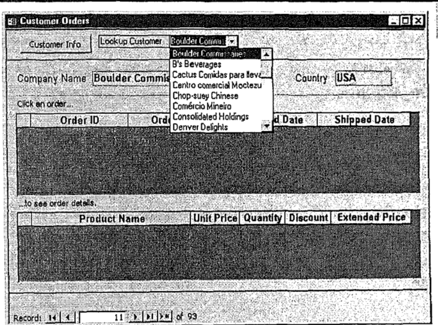
عند إضافة سجل جديد باستخدام نموذج Customers، لا يعرض أكسس تلقائياً السجل الجديد في نموذج Customer Orders، لذلك لعرض هذا السجل يجب إعادة الاستعلام عن مربع التحرير والسرد وعن النموذج. سنقوم بإنشاء إجراء حدث يعيد الاستعلام عن نموذج Customer Orders ومربع التحرير والسرد cboFind بمجرد الانتهاء من حفظ السجل الجديد، وسيتعرف نموذج Customers على حدث AfterInsert. ويكون هذا الإجراء كالتالي:

```
Private Sub Form_AfterInsert()
    Dim frm As Form, cbo As ComboBox
    If IsLoaded("Customer Orders") Then
        Set frm = Forms![Customer Orders]
        Set cbo = frm!cboFind
        DoCmd.SelectObject acForm, "Customer Orders"
        RunCommand acCmdSaveRecord
        cbo.Requery
        frm.Requery
    End If
End Sub
```

ما يثير الاهتمام في هذا الإجراء هو أنه يتم تشغيله من نموذج Customers، لكنه يتخذ الإجراءات من نموذج Customer Orders. وتتطلب هذه الإجراءات أن يكون نموذج Customer Orders مفتوحاً، وهو الأمر الذي يجب أن يحدده إجراء الحدث Form_AfterInsert قبل اتخاذ أي إجراءات، فيستخدم وظيفة IsLoaded ليوضح ما إذا كان نموذج Customer Orders مفتوحاً أو ينهيها إذا كان النموذج مغلقاً. فإذا كان النموذج مفتوحاً يستخدم الإجراء طريقة SelectObject لكائن DoCmd لتحديد نموذج Customer Orders ثم يقوم بتشغيل طريقة RunCommand لحفظ السجل في نموذج Customer Orders "يجب حفظ السجل قبل تشغيل طريقة Requery" وسيعيد الإجراء الاستعلام عن مربع التحرير والسرد ثم عن النموذج وذلك بتشغيل طريقة Requery لكل كائن.

افتح نموذج Customer في عرض Design وادخل الإجراء Form_AfterInsert كإجراء الحدث لحدث After Insert للنموذج ثم احفظ واغلق النموذج.

والآن ادخل Boulder Commissaries كاسم لعميل جديد في مربع التحرير والسرد في نموذج Customer Orders ثم اضغط Enter انقر yes لإدخال عميل جديد انقر زر Data Entry Mode. ادخل BOULD على أنه CustomerID واختر USA في مربع التحرير والسرد Country ثم ادخل أي بيانات أخرى عن عميل جديد في نموذج Customers واغلق نموذج Customers. وأخيراً، اسدل مربع التحرير والسرد Customer Orders وسيكون اسم العميل الجديد ضمن القائمة وبتحديده، سيظهر معلومات عن هذا العميل الجديد في النموذج "انظر الشكل ١٢-١٠".



الشكل ١٢-١٠ يعرض النموذج Customer Orders تلقائياً العميل الجديد بعد إضافته إلى نموذج Customers.

تحرير البيانات في مجموعة السجلات

على الرغم من أنك عادة ما تقدم النماذج في واجهة أكسس حتى يستطيع المستخدم تحرير البيانات، فإنه أحياناً يكون من الأفضل بل ومن الأسرع العمل مباشرة مع مجموعة السجلات بدلاً من العمل مع نموذج. ولقد تعرفت في فصول سابقة على كيفية التنقل في مجموعة السجلات من سجل إلى آخر وذلك بنقل مؤشر السجل الحالي. إذا كانت مجموعة السجلات من نوع الجدول أو من نوع المجموعة الحيوية قد تتمكن من تحرير السجلات الموجودة بالفعل أو من إضافة سجلات جديدة أو حذف سجلات أخرى وبالتقنيات الموضحة هنا، فإننا نفترض أنك تستطيع القيام بالتغييرات اللازمة لمجموعة السجلات.

ملاحظة تعتمد قدرتك على تغيير البيانات في مجموعة السجلات على عدد من العوامل مثل الخيارات التي قمت بإعدادها عند إنشاء مجموعة السجلات، نوع الاستعلام أو عبارة SGL التي تستخدمها مع مجموعة السجلات من نوع المجموعة الحيوية، كما تعتمد أيضاً على ما إذا كان قد قام مستخدمون آخرون بوضع عناصر مؤمنة تمنعك من إحداث أي تغييرات "إذا كنت تعمل في جو به مستخدمون عديدون". وبالاعتماد على طريقة تصميم الاستعلام قد تستطيع تحرير حقول معينة دون غيرها "انظر Queries و Results و Updating في التعليمات الفورية". غير أنه لا يمكنك تحرير مجموعة سجلات تستند إلى استعلامات جدولية أو توحيدية.

تغيير السجل

القاعدة الأساسية التي تتبع عند العمل مع مجموعة السجلات هي أنه لا يمكنك العمل فقط مع السجل الحالي، وهذا يعني أنه يجب عليك نقل مؤشر السجل الحالي إلى سجل آخر قبل البدء في التحرير. وكما ذكر سابقاً في الفصل السادس، يستخدم Jet موضعاً منفصلاً في الذاكرة يعرف باسم المخزن المؤقت المنسوخ ، وهو مخصص لمحتويات السجل التي يتم تحريرها. لتغيير سجل ستحتاج أولاً إلى نقل نسخة من السجل الحالي في المخزن المؤقت المنسوخ مستخدماً طريقة Edit، ثم عمل التغييرات اللازمة وحفظها في المخزن المؤقت المنسوخ إلى السجل الحالي مستخدماً طريقة Update "أو قم فقط بإفراغ المخزن المؤقت المنسوخ دون حفظ التغييرات وذلك باستخدام طريقة CancelUpdate".

إذا حاولت تحرير سجل دون نقله إلى المخزن المؤقت المنسوخ باستخدام طريقة Edit، سيظهر خطأ وقت التشغيل. أما إذا انتقلت إلى سجل آخر دون حفظ التغييرات إلى السجل الحالي باستخدام طريقة Update ينتج أي خطأ لكن لن تنقل التغييرات إلى السجل الحالي كما أن التغييرات التي قمت بعملها في المخزن المؤقت المنسوخ ستتعرض للضياع، إذا قمت بإغلاق مجموعة السجلات أو بإعداد خاصية Bookmark أو باستخدام طريقة Edit أو AddNew مرة أخرى، دون أن تكون قد استخدمت طريقة Update أولاً.

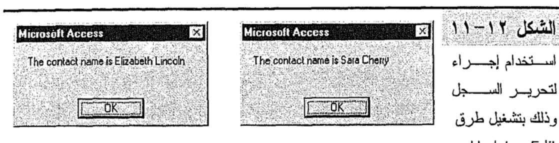
تلميح يمكن استخدام خاصية مجموعة السجلات EditMode لتحديد حالة التحرير بالنسبة للسجل الحالي. وتعيد هذه الخاصية قيماً بأعداد صحيحة تطابق الحالات التالية dbEditNone إذا لم يتم إحراز أي تقدم في عملية التحرير dbEditInProgress إذا كان قد تم استبعاد طريقة Edit وكانت هناك نسخة من السجل الحالي في المخزن المؤقت المخزون وdbEditAdd إذا كان قد تم استبعاد طريقة AddNew وكان المخزن المؤقت المنسوخ يحتوي على بيانات خاصة بسجل جديد ولم يتم حفظها في مجموعة السجلات

وكمثال يجد إجراء EditRecordset السجل الخاص بالعميل Bottom-Dollar Markets في جدول Customers ويغير اسم جهة الاتصال.

```
Public Sub EditRecordset()
   Dim rst As Recordset
   Set rst = CurrentDB.OpenRecordset("Customers")
   rst.Index = "PrimaryKey"
   rst.Seek "=", "BOTTM"
   If rst.NoMatch Then
      MsgBox "There is no customer with this CustomerID"
      Exit Sub
   Else
      MsgBox "The contact name is " & rst!ContactName
      rst.Edit
      rst!ContactName = "Sara Cherry"
      rst.Update
      MsgBox "The contact name is " & rst!ContactName
   End If
End Sub
```

يفتح هذا الإجراء مجموعة سجلات من نوع الجدول في جدول Customers، كما يقوم بإعداد الفهرس الحالي للفتاح الأساسي وباستخدام طريقة Seek ليتخذ موضعاً للسجل في Bottom-Dollar Markets في حالة عدم أيجاد السجل، سيعرض الإجراء رسالة ثم ينتهي بعد ذلك. أما في حالة إيجاده سيعرض الإجراء اسم جهة الاتصال الخاص بالسجل في المخزن المؤقت المنسوخ، ثم تغيير اسم جهة الاتصال ثم تشغيل طريقة Update لحفظ السجل المتغير في جدول، وأخيراً عرض اسم جهة الاتصال المتغير.

قم بإنشاء وحدة نمطية قياسية جديدة تحمل اسم basRecordset ثم ادرج إجراء EditRecordset المذكور سلفاً. قم بعد ذلك بتشغيل إجراء EditRecordset في نافذة Immediate وسيعرض الإجراء مربعات رسائل تحمل الاسم الحالي لجهة الاتصال يليه اسم جهة الاتصال المحرر "انظر الشكل ١٢-١١".



الشكل ١٢-١١

استخدام إجراء لتحرير السجل وذلك بتشغيل طرق Edit وUpdate لمجموعة السجلات. سيعرض الإجراء مربعات رسائل بها الاسم الحالي لجهة الاتصال "أ" يليه اسم جهة الاتصال المحرر "ب".

تحذير

هناك خطأ شائع وهو إغفال استبعاد طريقة Edit قبل محاولة تغيير البيانات في السجل، لحسن الحظ، ينتج عن هذا الخطأ خطأ وقت التشغيل فتنتبه إلى خطأك. كما أن هناك خطأ آخر بنفس درجة شيوع الخطأ الأول وهو إغفال استبعاد طريقة Update بعد الانتهاء من تغيير سجل أو من إدخال البيانات في سجل جديد، غير إن هذا الخطأ لا ينتج عنه الخطأ وقت التشغيل، لكن تجاهل محتويات المخزن المؤقت المخزن بدون تحذير. لذا، دون تلقي المساعدة من رسالة خطأ وقت التشغيل، تكون الأخطاء الناجمة عن الفشل في التحديث أكثر من صعوبة في تصحيحها "حلها".

إضافة سجل

لإضافة سجل جديد وحفظ التغييرات، يجب اتباع عملية تقوم على ثلاث خطوات:

١- قم بإنشاء سجل جديد في المخزن المؤقت المنسوخ ثم قم بإعداد أي قيم افتراضية مستخدماً طريقة AddNew فيقوم أكسس بإعداد تلك القيم التي حددتها في جدول العرض Design، وبإعداد قيم الحقول بدون قيماً افتراضية لتكون Null.

٢- قم بإدخال البيانات الجديدة.

٣- احفظ التغييرات التي قمت بعملها في المخزن المؤقت المنسوخ وأضف السجل المحفوظ إلى مجموعة السجلات مستخدماً ٣- طريقة Update "أو افرغ المخزن المؤقت المنسوخ بدون إضافة السجل الجديد مستخدماً طريقة CancelUpdate".

يعتمد موضع السجل الجديد في مجموعة السجلات على نوع مجموعة السجلات، فعند إضافة سجل إلى مجموعة السجلات من نوع الجدول، يكون موضعه في نهاية مجموعة السجلات، وذلك إذا لم تعد خاصية Index. فإذا قمت بإعداد الفهرس الحالي مستخدماً خاصية Index يتم إدراج السجل الجديد في مكانه الصحيح في ترتيب مصنف تبعاً للفهرس الحالي. أما عند إضافة السجل الجديد إلى مجموعة السجلات من نوع المجموعة الحيوية، فيكون موضعه في نهاية مجموعة السجلات. وفي أي من الحالتين يستمر مؤشر السجل الحالي في الإشارة إلى السجل الذي كان موجوداً من قبل، قبل إضافة السجل الجديد. لجعل السجل الجديد سجلاً حالياً، قم بإعداد خاصية Bookmark بحيث تكون LastModified.

في حالة الانتقال إلى سجل آخر بدون حفظ التغييرات باستخدام طريقة Update، لن ينشأ خطأ ما، لكن لن يتم إضافة السجل الجديد وستضيع التغييرات المخزنة في المخزن المؤقت المنسوخ والتي قد تضيع أيضاً عند إغلاق مجموعة السجلات أو إعداد خاصية Bookmark لسجل آخر، أو استخدام طريقة Edit أو AddNew مرة أخرى بدون استخدام طريقة Update أولاً. ويمكن كتابة إجراء VBA لاختبار خاصية السجل Dirty فإذا تم تقييم IsDirty "اسم النموذج" على أنه True، يمكن أن يعرض أكسس مربع رسالة ينبه المستخدم إلى إن التغييرات ستضيع، ويخبره بأنه يمكن تضمين عبارة If...Then لتلغي الإجراء وتسمح للمستخدم بحفظ السجل وذلك بنقر Yes في مربع الرسالة.

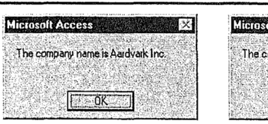
وكمثال على ذلك، يضيف إجراء AddRecordset سجلاً جديداً في جدول Customers:

```
Public Sub AddRecordset()
    Dim rst As Recordset
    Set rst = CurrentDB.OpenRecordset("Customers")
    rst.Index = "PrimaryKey"
```

```
    rst.AddNew
    rst!CustomerID = "AARDV"
    rst!CompanyName = "Aardvark Inc."
    rst!Country = "Australia"
    rst.Update
    rst.Bookmark = rst.LastModified
    MsgBox "The company name is " & rst!CompanyName
    rst.MoveNext
    MsgBox "The company name is " & rst!CompanyName
End Sub
```

يفتح هذا الإجراء مجموعة من السجلات من نوع الجدول في جدول Customers ثم يقوم بإعداد الفهرس الحالي للمفتاح الأساسي. يقوم هذا الإجراء بتشغيل طريقة AddNew للإنشاء سجل جديد في المخزن المؤقت المنسوخ وبإعداد قيم CustomerID وCompanyName وCountry fields وبتشغيل طريقة Update لحفظ السجل في الجدول. نظراً لأن الفهرس الحالي قد تم إعداده بالفعل، فإن السجل الجديد يتم إدراجه في ترتيب المفتاح الأساسي كالسجل الأول لمجموعة السجلات. كما يقوم هذا الإجراء بإعداد خاصية Bookmark بحيث تكون LastModified وذلك لنقل مؤشر السجل الحالي إلى السجل التالي ثم يعرض اسم الشركة الخاص بالسجل التالي في مجموعة السجلات.





الشكل ١٢-١٢

يفتح إجراء Recordset مجموعة سجلات في جدول Customers ويضيف سجلاً آخر جديداً في بداية مجموعة السجلات "أ" كما يعرض معلومات عن السجل الثاني "ب".

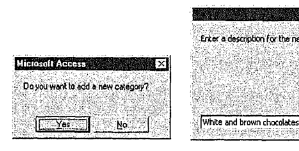
وكمثال آخر، سنضيف فئة جديدة إلى مربع التحرير والسرد الخاص بالفئة في نموذج Products بدون فتح نموذج Categories وعن مصدر مربع التحرير والسرد الخاص بالفئة، يمكن القول بأنه يكون جدول Categories، لذلك، فإضافة قيمة جديدة إلى قائمة التحرير والسرد يتطلب إضافة سجل جديد في جدول Categories، ويتناول إجراء الحدث CategoryID_NotInList تلك الإضافة.

```
Private Sub CategoryID_NotInList (NewData As String, Response _
    As Integer)
   Dim intNew As Integer, strDescription As String, rst As Recordset
   intNew = MsgBox("Do you want to add a new category?", vbYesNo)
   If intNew = vbYes Then
      Set rst = CurrentDB.OpenRecordset("Categories")
      rst.AddNew
      rst!Categoryname = NewData
      strDescription = InputBox("Enter a description for the new " _
         & "category.")
      rst!Description = strDescription
      rst.Update
      Response = acDataErrAdded
   Else
      MsgBox "The value you entered is not a valid category"
      RunCommand acCmdUndo
      Response = acDataErrContinue
   End If
End Sub
```

يعرض هذا الجزء مربع رسالة يسأل ما إذا كان المستخدم يريد إضافة فئة جديدة، ففي حالة نقر زر yes، يفتح الإجراء مجموعة سجلات على جدول Categories ويطلب من المستخدم إدخال وصفاً للفئة الجديدة، ثم يضيف منتجاً جديداً مباشرةً إلى الجدول ويعد وسيطة Response لأن تكون acDataErrAdded وذلك لإعطاء تعليمات لأكسس بإعادة استعلام قائمة التحرير والسرد. أما في حالة نقر زر No، فإن الإجراء يعرض رسالة مخصصة ثم يتراجع عن الإدخال ويوقف رسالة الخطأ الافتراضية.

لاختبار هذا الإجراء، حدد نموذج Products في نافذة Database وانقر زر Code في شريط الأدوات. قم بعد ذلك بإنشاء إجراء CategoryID_NotInList ثم احفظ النموذج وانتقل

إلى عرض Form انقر زر New لعرض سجل جديد. اكتب Chocolates في قائمة التحرير والسرد Category ثم اضغط Enter. انقر زر Yes لإدخال فئة جديدة (انظر الشكل ١٢-١٣ أ) واكتب White and brown chocolates في مربع الإدخال (انظر الشكل ١٢-١٣ب) ثم انقر Ok حينئذ سيضيف أكسس الفئة الجديدة إلى جدول Categories وسيعيد الاستعلام عن قائمة التحرير والسرد ثم يعرض الفئة الجديدة.



الشكل ١٢-١٣

إضافة قيمة جديدة إلى قائمة التحرير والسرد وذلك بإضافة سجل إلى الجدول الذي يعيد مصدراً لبيانات القائمة.

حذف سجل

يحذف السجل في خطوة واحدة، فقط احذف السجل الحالي باستخدام طريقة Delete، غير أنه لا يوجد مخزن مؤقت لإبقاء المحتويات في حالة العدول عن هذه الخطوة، فالحذف يكون فورياً ولا يمكن الرجوع فيه. ويظل السجل المحذوف هو السجل الحالي، على الرغم من أنك قد لا تشير إلى سجل حالي صالح إذا كنت تنوي استبعاد أي طرق تتطلب سجل حالي صالح.

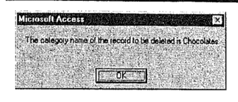
وكمثال على ذلك، يحذف إجراء DeleteRecordset السجل الذي تم إضافته إلى جدول Categories في المقطع الأخير.

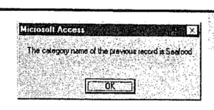
```
Public Sub DeleteRecordset()
   Dim rst As Recordset
   Set rst = CurrentDB.OpenRecordset("Categories", dbOpenDynaset)
   rst.FindFirst "CategoryName = 'Chocolates'"
   MsgBox "The category name of the record to be deleted is " & _
      rst!CategoryName
   rst.Delete
   rst.MovePrevious
```

```
    MsgBox "The category name of the previous record is " & _
        rst!CategoryName
End Sub
```

يفتح هذا الإجراء مجموعة سجلات من نوع المجموعة الحيوية في جدول Categories، كما يستخدم طريقة FindFirst لنقل مؤشر السجل الحالي إلى السجل ثم يشغل طريقة Delete لحذف السجل.

ادرج إجراء DeleteRecordset في وحدة basRecordset النمطية، ثم قم بتشغيله في نافذة Immediate، وسيعرض الإجراء مربعات الرسائل موضحاً السجل الذي سيتم حذفه، "انظر الشكل ١٢-١٤ أ" والسجل السابق الذي أصبح هو السجل الحالي عند قيام الإجراء بتشغيل طريقة MovePrevious بعد حذف السجل "انظر ١٢-١٤ب".





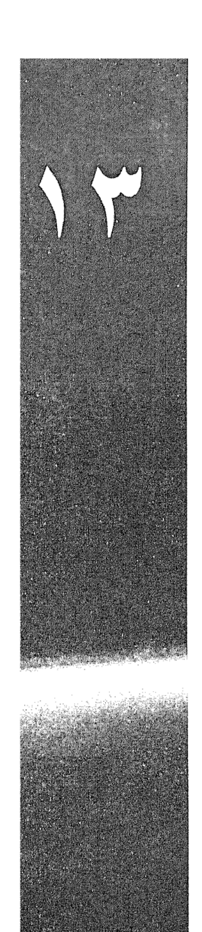
الشكل ١٢-١٤
رسائل قبل وبعد تشغيل طرق Delete "أ" وmovePrevious "ب".

خلاصة

كان التركيز في هذا الفصل على استخدام إجراءات VBA للعمل مع إجراءات البيانات بطريقتين. في طريقة النماذج، تستخدم النماذج للعمل مع البيانات. كما غطى هذا الفصل الإجراءات التي لا تستخدم لأتمتة عمليات إضافة وتحرير وحذف السجلات في نموذج ما. أما في طريقة مجموعة السجلات فيتخذ المستخدم بعض الإجراءات التي تقوم بتشغيل إجراءات أخرى تقوم بتحرير أو حذف أو إضافة سجلات جديدة مباشرةً في مجموعة السجلات بدون عرض النموذج في. ومن النقاط الهامة التي وردت في هذا الفصل:

- يمكن الحفاظ على البيانات من التغييرات غير المقصودة وذلك باستخدام نموذج به أزرار أوامر تقوم بتشغيل الإجراءات للتبديل بين أوضاع المراجعة وإدخال البيانات.
- يمكن استخدام الإجراءات لتخصيص صلاحية البيانات، بتغيير مثلاً توقيت اختبار الصلاحية.

- يمكن استخدام Command Button Wizard لإنشاء إجراءات بسيطة للمحافظة على البيانات، وتعديل الإجراءات لتجنب أخطاء وقت التشغيل ولإيقاف رسائل الخطأ الافتراضية.
- يمكن جعل إدخال البيانات أكثر كفاءة وذلك بأخذ الترتيبات اللازمة لنقل القيم إلى سجل جديد بإحدى طريقتين: إما باستخدام إجراء لإعداد خاصية DefaultValue أو باستخدام إجراء لنقل قيم الحقل من سجل إلى سجل جديد.
- يمكن استخدام طريقة Requery لعرض البيانات الأكثر حداثة، وذلك في حالة، وذلك في حالة عدم تحديث أكسس للنموذج تلقائياً.
- يمكن استخدام طرق Edit وAddNew وDelete الخاصة بكائن Recordset للعمل مباشرة مع مجموعة سجلات في الذاكرة بدون فتح النموذج.



العمل مع مجموعة من السجلات باستخدام VBA أكسس

تعلمت في الفصلين السابقين كيفية إنشاء إجراءات للعمل مع سجل واحد باستخدام طريقتين مختلفتين وهما النماذج ومجموعات السجلات. في طريقة النماذج، يعمل المستخدم مع النماذج للتنقل بين السجلات والتحكمات لاستعراض وتعديل البيانات. آما في طريقة مجموعات السجلات، تقوم إجراءات المستخدم بتشغيل إجراءات تنشئ وستغل مجموعة السجلات في الذاكرة دون عرض مرئي. يركز هذا الفصل على العمل مع مجموعات من السجلات باستخدام الطريقتين:

يشرح الجزء الأول من هذا الفصل كيفية استخدام طريقة النماذج لأداء المهام التالية

- جعل العمليات لفرز السجلات ولتحديد مجموعة من السجلات التي تفي بمطلب معايير البحث باستخدام نموذج أو تقرير.
- فتح نموذج أو تقرير يعرض تحديدا محدودا للسجلات عند الفتح.
- تغيير التحديد بعد فتح النموذج أو التقرير.
- استخدام نموذج لتجميع معايير البحث وتمرير المعايير لاستعلام "Query By Form".
- استخدام مربع قائمة متعدد التحديد لتحديد السجلات.

يركز الجزء الثاني من الفصل على العمل مع مجموعات السجلات في إجراء VBA ستعلم كيفية استخدام الاستعلامات المخزنة وعبارات SQL لفرز وتحديد مجموعة من السجلات وكيفية استخدام استعلامات لإجراء تغييرات في مجموعة السجلات.

فرز السجلات في نموذج أو تقرير

يمكن جعل عملية فرز السجلات في نموذج أو تقرير تلقائية عن طريق إعداد خصائص OrderBy و OrderBy On بالنموذج أو التقرير في إجراء.

ملاحظة للحصول على خبرة في متناول اليد مع التقنيات التي سيتم شرحها في هذا الفصل، قم بإنشاء نسخة جديدة من نموذج قاعدة البيانات Northwind تدعى Northwind Ch 13 واعمل خلال خطوات الأمثلة.

نبدأ بإنشاء نموذج جديد لقاعدة بيانات Northwind CH_13 والتي ستستخدمها لتوضيح وإجراءات فرز وتصفية النموذج

١- قم بإنشاء نموذج جديد باستخدام Form Wizard. حدد الحقول التالية:

Orders table	OrderID, OrderDate, RequiredDate, and ShippedDate
Shippers table	CompanyName
Customers table	CompanyName
Employees table	LastName

٢- انقر Next. أختر لاستعراض البيانات عن طريق Orders أخـــتر عــرض Tabular ونمط Standard قم بتسمية النموذج frmOrder Status.

٣- افتح النموذج في أسلوب عرض Design عين خاصية Caption الخاصة بالنموذج إلى Order Status وخاصية Allow Additions إلى No."نمـوذج Order Status هـو نموذج مراجعة وليس معدا لإدخال بيانات وإعدادات خاصية Allow Additions إلـــى No يخفي السجل الخالي في نهاية كل مجموعة سجلات".

٤- حدد كل تحكمات البيانات وعين خاصيــة Locked الخاصـــة بــها إلـــى Yes وعيــن خصائص التنسيق كما هو موضح بالأسفل ولأن النموذج ليس معدا لإدخال بيانات، نقـوم بإغلاق التحكمات لتجنب التغيرات غير المقصودة:

BackColor	12632256
SpecialEffect	Flat
BorderStyle	Transparent

ملاحظـــة

إذا عينت خاصية Allow Edits الخاصة بالنموذج إلى No، لا تســـمح أي من التحكمات بأية تغييرات. وبتعيين خاصية Locked للتحكمات الفرديـــة، تسمح بالتغييرات "التحريرات" لقمة البحث في مربع السرد والتحرير لذلـــك يستطيع المستخدمون استخدام مربع سرد وتحرير فـــي النمـــوذج لاختيـــار السجلات.

٥- رتب التحكمات بالترتيب التالي وغير خاصية Caption كما هو مبيـــن غـــير خاصيـــة Font Weight للعناوين إلى Bold. عندما تنتهي من عملك، يجب أن يكـــون النمـــوذج مثل ذلك الموجود في شكل "١٣-١":

Caption	**ControlName**
Order Date	OrderDate
Required Date	RequiredDate
Shipped Date	ShippedDate
Shipper	Shippers.CompanyName
Customer	Customers.CompanyName
Employee	LastName

٦- اختر File ⇦ Save As واحفظ نسخة من النموذج بالاسم الجديد frmOrderStatusClear

Order Status

Order ID	Order Date	Required Date	Shipped Date	Company Name	Company Name	Last Name
[illegible]	28-Jul-1997	25-Aug-1997	01-Aug-1997	United Package	Save-a-lot Markets	Davolio
10542	20-May-1997	17-Jun-1997	26-May-1997	Federal Shipping	Königlich Essen	Davolio
10275	07-Aug-1996	04-Sep-1996	09-Aug-1996	Speedy Express	Magazzini Alimentari Riuniti	Davolio
10746	19-Nov-1997	17-Dec-1997	21-Nov-1997	Federal Shipping	Chop-suey Chinese	Davolio
10827	12-Jan-1998	26-Jan-1998	06-Feb-1998	United Package	Bon app'	Davolio
10655	03-Sep-1997	01-Oct-1997	11-Sep-1997	United Package	Reggiani Caseifici	Davolio
10579	25-Jun-1997	23-Jul-1997	04-Jul-1997	United Package	Let's Stop N Shop	Davolio
10314	25-Sep-1996	23-Oct-1996	04-Oct-1996	United Package	Rattlesnake Canyon Grocery	Davolio
10304	12-Sep-1996	10-Oct-1996	17-Sep-1996	United Package	Tortuga Restaurante	Davolio
10546	23-May-1997	20-Jun-1997	27-May-1997	Federal Shipping	Victuailles en stock	Davolio
10995	02-Apr-1998	30-Apr-1998	06-Apr-1998	Federal Shipping	Pericles Comidas clásicas	Davolio
10270	01-Aug-1996	29-Aug-1996	02-Aug-1996	Speedy Express	Wartian Herkku	Davolio
10616	31-Jul-1997	28-Aug-1997	05-Aug-1997	United Package	Great Lakes Food Market	Davolio
10626	11-Aug-1997	08-Sep-1997	20-Aug-1997	United Package	Berglunds snabbköp	Davolio
10630	13-Aug-1997	10-Sep-1997	19-Aug-1997	United Package	Königlich Essen	Davolio

Record: 1 of 830

الشكل ١٣-١ قم بإنشاء نموذج Order Status للفرز و التحديد.

وضع السجلات في ترتيب تنازلي أو ترتيب تصاعدي

يمكنك استخدام خاصية OrderBy لنموذج أو تقرير للفرز عن طريق حقل واحد أو لإنشاء فرز معقد عن طريق حقول متعددة بعضها في ترتيب تصاعدي والآخر في ترتيب تنازلي. وخاصية OrderBy هي تعبير سلسلة "مسلسل"يتكون من اسم الحقول أو الحقل الذي تريد فرزه مرتبة في ترتيب الفرز ومفصولة بفاصلات. والتعيين الافتراضي لخاصية OrderBy تصاعدي.

لفرز حقل في ترتيب تنازلي، تضمن الكلمة الأساسية DESC بعد اسم الحقل. على سبيل المثال، لفرز نموذج Frm Order Status باسم عميل ثم بتاريخ ترتيب في ترتيب تنازلي، عليك بتعيين خاصية OrderBy كما يلي: Customers.CompanyName, OrderDate DESC

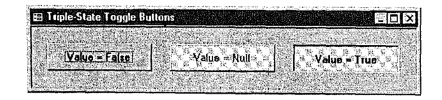
إذا لم تضمن DESC في نهاية عبارة تعيين خاصية OrderBy. يسرد النموذج أو التقرير السجلات في الترتيب التصاعدي للحقل الذي قمت بتحديده.

يحدد تعيين خاصية OrderBy ترتيب فرز جديد لكن لا يقوم بالفرز. تعيين خاصية OrderBy On إلى True أو False لتطبيق أو إزالة الفرز الذي حددته لنموذج ما، يمكن تعيين خاصية OrderBy في ورقة خاصية النموذج أو في إجراء VBA ثم استخدام إجراء VBA لتطبيق وإزالة الفرز باستخدام خاصية OrderBy On ولتقرير ما، يمكنك تعيين ما في خاصيتي OrderBy وOrderBy On في ورقة خاصية التقرير أو في إجراء VBA.

استخدام زر تبديل ثلاثي الحالة

لتوضيح هذه الأفكار، سنقوم بإنشاء وإجراء حدث لفرز سجلات نموذج Order Status بعميل "لعميل" سنستخدم زر تبديل بدلا من زر أمر لأن زر التبديل له قيمة يمكن استخدامها عن طريق إجراء فرضيا، لزر التبديل قيمتان وهما True وfalse لكن تعيين خاصية Triple State إلى Yes يعطي قيمة ثالثة تساوي Null

- ♦ عندما يكون لزر التبديل قيمة False، يبدو مرفوعا كزر الأمر.
- ♦ نقر الزر يغيره إلى حالة Null وعندما يكون الزر مستويا لكن لا يبدو غاطسا."عندما يكون الزر مستويا وليس غاطسا"
- ♦ نقر الزر مرة أخرى يغير إلى حالة True حيث يبدو الزر المستوي غاطسا.



يختبر إجراء Tgl Customer Click حالة الزر ويزيل الفرز إذا كان الزر في حالة False ويطبق فرز تصاعدي إذا كان الزر في حالة Null ويطبق فرز تنازلي إذا كان الزر في حالة True:

```
Private Sub tglCustomer_Click""
    Dim str As String, tgl As ToggleButton
    str = "Customers_CompanyName"
    Set tgl = tglCustomer
    Select Case tgl.Value
        Case True
```

```
            Me.OrderBy = str & " DESC"
            Me.OrderByOn = True
        Case False
            Me.OrderByOn = False
        Case Else
            Me.OrderBy = str
            Me.OrderByOn = True
    End Select
End Sub
```

ينشئ هذا الإجراء متغير Str لحمل اسم حقل الفرز ومتغير الكائن Tgl للإشارة إلى زر التبديل يستخدم إجراء الحدث خاصية Me للإشارة إلى النموذج لاختبار حالة زر التبديل، يستخدم الإجراء بنية Select Case "تبادليا يمكنك استخدام بنية "If...Then...Else.

- ♦ إذا كان لزر التبديل قيمة True يتم تعيين خاصية OrderBy للنموذج إلى فرز تنازلي عن طريق Company Name ويتم تشغيل خاصية OrderBy On الخاصة بالنموذج.
- ♦ إذا كان لزر التبديل قيمة False يتم إيقاف خاصية OrderBy On الخاصة بالنموذج بحيث يتم إعادة السجلات لترتيب لم يتم فرزة.
- ♦ إذا لم يكن لزر التبديل قيمة True أو قيمة False يتم تعيين خاصية OrderBy الخاصة بالنموذج لفرز تصاعدي عن طريق Company Name ويتم تشغيل خاصية OrderBy On.

اتبع هذه الخطوات لإضافة زر التبديل لنموذج Order Status.

١- بدل لأسلوب عرض DesignK حدد كل التحكمات في قسم التفاصيل وعين خصائص Enabled إلى No لا تحتاج التحكمات إلى كونها متاحة عند استخدامها لخاصية OrderBy للفرز.

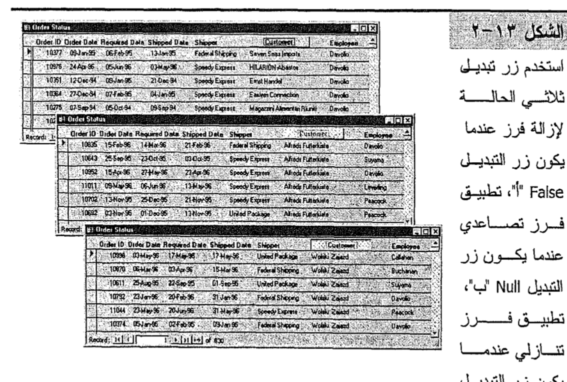
٢- أحذف بطاقة عنونة Customer في قسم العنوان واستبدلها تبديل، عين الخصائص كما يلي:

Name	**TglCustomer**
Caption	Customer
Default Value	False
TripleState	Yes

Name	**TglCustomer**
FontWeight	Bold
ControlTip Text	Click to toggle no sort, ascending, descending

٣- انقر في خاصية حدث OnClick وانقر زر Build يمين مربع الخاصيـــة ادرج إجـــراء حدث Tgl Customer Click الموضـح سابقا.

٤- احفظ النموذج. بدل لأسلوب عرض Form وانقر زر التبديل. بينما تتجول في الحالات، يتم فرز السجلات باسم العميل في ترتيب تصاعدي أولا ثم يتم فرزها في ترتيب تنـازلي ثم تتم إزالة الفرز. "انظر شكل ١٣-٢".



الشكل ١٣-٢

استخدم زر تبديـل ثلاثـي الحالــة لإزالة فرز عندما يكون زر التبديـل False "أ"، تطبيـق فـــرز تصاعـــدي عندما يكـــون زر التبديل Null "ب"، تطبيـــق فــــرز تنـــازلي عنــــدما يكون زر التبديـل True "ج".

تزامن تعيينات الفرز

لأن Access يحفظ تعيينات OrderBy وOn OrderBy الفعالة عندما تغلق النمـــوذج ولكنـة لا يحفظ حالة زر التبديل، يمكن أن تتخلص الحالات الثلاثة من التزامن مع بعضها البعض عنـــدما تفتح النموذج لأول مرة. يبدأ إجراء حدث From Open الموضح بالأســفل خصائـــص الفـــرز

بتعيين خاصية OrderBy لسلسلة طولها صفر وخاصية OrderBy On إلى False عندما يفتح النموذج:

```
Private Sub Form_Open"Cancel As Integer"
    Me.OrderBy = """"
    Me.OrderByOn = False
End Sub
```

لإضافة هذا الإجراء، بدل لأسلوب عرض Design، انقر في خاصية حدث النموذج On Open وانقر زر Build في يمين مربع الخاصية. ادرج إجراء الحدث Form Open في وحدة النموذج. احفظ وأغلق النموذج.

والآن افتح النموذج، يتم بدء كلا من تعيينات خاصية الفرز وهما متزامنان مع زر التبديل. وبينما تتجول عبر الحالات، يتم فرز السجلات باسم العميل أولا في ترتيب تصاعدي ثم فرزها في ترتيب تنازلي ثم تتم إزالة الفرز.

الفرز بأي عمود

عندما يتم عرض السجلات في نموذج مجدول، يمكنك جعل فرز السجلات تلقائي بأعمدة محددة. استبدل بطاقة عنونة كل عمود تريد فرزه بزر تبديل ثلاثي الحالة وقم بإنشاء إجراء حدث مثل إجراء Tgl Customer Click، لفرز السجلات بحقل في العمود باستخدام هذه التقنية، كل من الفروز مستقل، ويتجاوز آخر زر تبديل تم نقرة الفروز السابقة ويحدد الفرز الأخير، على سبيل المثال، إذا بدلت زر Customer للفرز تصاعديا بالعميل ثم تبدل زر Employee للفرز تصاعديا بالموظف، يتم تخزين السجلات في ترتيب تصاعدي عن طريق الموظف

كمثال سنضيف زر تبديل لفرز عمود Employee الخاص بنموذج Order Status.

١ – بدل إلى أسلوب عرض Design، احذف بطاقة عنونة Employee ثم استبدلها بزر تبديل عين خاصية Name في Tgl Employee وخاصية Caption في Employee عين الخصائص الأخرى حتى تكون مثل تلك التي قمت بتعيينها لزر تبديل Tgl Customer.

٢ – انقر زر Code في شريط الأدوات لفتح وحدة النموذج. حدد إجراء Tgl Customer Click وانسخ الإجراء. انقر في حدث OnClick الخاص بزر التبديل الجديد، انقر زر Build ثم الصق الإجراء في إطار Module. غير عبارة التعيين للمتغيرات Str وTgl كما يلي:

```
str = "LastName"
Set tgl = tglEmployee
```

٣- احفظ النموذج وبدل لأسلوب عرض Form.

٤- انقر زر تبديل Customer ثم انقر زر تبديل Employee، يتم تخزين السجلات الآن عن طريق الموظف، "انظر شكل ١٣-٣".

Order Status

Order ID	Order Date	Required Date	Shipped Date	Shipper	Customer	Employee
10372	04-Jan-95	01-Feb-95	09-Jan-95	United Package	Queen Cozinha	Buchanan
10648	28-Sep-95	09-Nov-95	10-Oct-95	United Package	Ricardo Adocicados	Buchanan
10358	21-Dec-94	18-Jan-95	28-Dec-94	Speedy Express	La maison d'Asie	Buchanan
10650	29-Sep-95	27-Oct-95	04-Oct-95	Federal Shipping	Familia Arquibaldo	Buchanan
10269	31-Aug-94	14-Sep-94	09-Sep-94	Speedy Express	White Clover Markets	Buchanan
10721	29-Nov-95	27-Dec-95	01-Dec-95	Federal Shipping	QUICK-Stop	Buchanan
10359	22-Dec-94	19-Jan-95	27-Dec-94	Federal Shipping	Seven Seas Imports	Buchanan
10378	10-Jan-95	07-Feb-95	19-Jan-95	Federal Shipping	Folk och fä HB	Buchanan

Record: 1 of 830

الشكل ١٣-٣
يمكن تخزين النموذج عن طريق عمود Customer أو Employee

القيام بفرز ذي طبقتين

مع الفرز عن طريق العمود، يمكنك القيام بفرز ثنائي الطبقة لكل من مع وجود تحديدات. كمثال، يختبر إجراء Set OrderBy قيمة زر تبديل Customers ثم يؤدي إلى "فرز فرعي" عندما يتم نقر زر تبديل Employee معتمدا على قيمة زر Customers لهذا يعني إن الفرز ثنائي الطبقة يعمل فقط بوجود حقل Customers كطبقة أولى "إذا تم تحديده":

```
Private Sub SetOrderBy""
    Const conCustomers = "Customers_CompanyName"
    Const conEmployees = "LastName"

    Dim strSortCustomers As String
    Dim strSortEmployees As String
    Dim strSort As String

    Select Case tglCustomer.Value
        Case True
            strSortCustomers = conCustomers & " DESC"
        Case False
            strSortCustomers = ""
        Case Else
```

```
      strSortCustomers = conCustomers
End Select

Select Case tglEmployee.Value
   Case True
      strSortEmployees = conEmployees & " DESC"
   Case False
      strSortEmployees = ""
   Case Else
      strSortEmployees = conEmployees
End Select

If strSortCustomers = "" Then
   If strSortEmployees = "" Then
      ' No sort on either field
      strSort = ""
   Else
      ' Employees is the only sort
      strSort = strSortEmployees
   End If
Else
   If strSortEmployees = "" Then
      ' Customers is the only sort
      strSort = strSortCustomers
   Else
      ' Both fields are sorted
      strSort = strSortCustomers & ", " & strSortEmployees
   End If
End If

If strSort = "" Then
   Me.OrderByOn = False
Else
   Me.OrderBy = strSort
```

```
        Me.OrderByOn = True
    End If
End Sub
```

يعمل هذا الإجراء بتدقيق زر Customers مدققا حالة زر Employees ثم إنشاء سلسلة فرز معتمدة على القيم الخاصة بهذه الحالات.

يبدأ الإجراء بإنشاء ثوابت "conCustomers, conEmployees" لتمثيل الحقول التي ستقوم الأزرار بفرزها ثم ينشئ سلاسل لقيم أزرار التبديل ,"StrSortEmployees, "StrSort,Customers وسلسلة الفرز النهائية "Str Sort"عندما تنقر أي من زري Customer أو Employee، يستخدم الإجراء المجموعة الأولى من عبارات Select Case لاختبار حالة زر Customer. ثم يختبر قيمة زر Employee في وظيفة Select Case التالية. بعد فرز قيمة زري الفرز، يعين الإجراء قيمة لمتغير StrSortمعتمدة على قيم المتغيرات StrSortEmployees وStrSort Customers.

- إذا لم يتم تحديد أيا من زر Customer أو زر Employee "كلا منهما له قيمة "False يعين الإجراء متغير StrSort كسلسلة طولها صفر والسجلات لم يتم فرزها.
- إذا تم تحديد زر Employee ولم يتم تحديد زر Customer يقوم الإجراء بفرز السجلات عن طريق Employee سواء تصاعديا أو تنازليا معتمدا على ما إذا كانت قيمة زر Employee هي True أو Null.
- إذا تم تحديد Customer ولم يتم تحديد زر Employee يقوم الإجراء بفرز السجلات عن طريق Customer سواء تصاعديا أو تنازليا معتمدا على ما إذا كانت قيمة زر Customer هي True أو Null.
- إذا تم تحديد كلا من زري Customer وemployee، يتم تعيين قيمة إلى StrSort تساوي فرز Customer "StrSort Customer" متبوعة بفرز "StrSort Employee "Employee يتم فصل هذه المتغيرات بفاصلة ومسافة "," وهذا هو التنسيق القياسي لسلسلة الفرز.

ولأن الإجراء متماثل سواء تم نقر زر Customer أو زر Employee يمكننا إنشاء إجراء مستوى نموذج وتغيير خاصية OnClick بالزرين لهذا الإجراء، ادخل إجراء Set OrderBy الموضح سابقا في وحدة النموذج. افتح نموذج frm Order Status Clean وغير خاصية OnClick لكل من الزرين إلى "" =Set OrderBy. احفظ النموذج وبدل لأسلوب عرض Form. اختبر الفرز في نموذج Order Status. على سبيل المثال انقر زر Customer لفرز السجلات عن طريق العميل ثم انقر زر Employee لفرز السجلات لكل عميل عن طريق الموظف.

تحديد مجموعات من السجلات في نموذج أو تقرير

من العمليات الشائعة والمهمة جدا في تطبيق قاعدة البيانات هو تحديد مجموعة من السجلات تفي بشروط البحث. على سبيل المثال، في إدخال ترتيب قاعدة البيانات ربما تريد تصميم نموذج يعرض ترتيبات لعميل بعينة أو كل الترتيبات الموضوعة بعد تاريخ محدد أو الترتيبات لعميل محدد موضوعة بعد تاريخ محدد وتم التعامل معها من قبل موظف محدد. تقوم أنت بتحديد السجلات من مجموعة السجلات الكاملة عن طريق تحديد شروط بحث تستخدمها لتصفية السجلات التي تفي بالشروط. يوفر Access تقنيات عديدة لتصفية السجلات.

شروط البحث

الخطوة الأولى في تحديد مجموعة من السجلات هي إعداد شروط البحث يمكنك استخدام شروط البحث الخمسة التالية:

اختبار المقارنة Comparison Test: يقارن قيمة تعبير بقيمة تعبير آخر. على سبيل المثال، يبحث Order Date < #1/1/99# عن ترتيبات في جدول Orders الموجود قبل 1/1/99.

اختبار المجال Range Test: يختبر ما إذا كانت قيمة تعبير تقع بين مجال قيم أم لا. على سبيل المثال، يبحث Order Date Between #9/1/98# And 1/1/99 عن ترتيبات في مجال البيانات المحدد.

عضوية في مجموعة Membership in a group: يختبر ما إذا كانت قيمة تعبير ما تلائم أخرى في مجموعة من القيم. على سبيل المثال، يبحث Country in"France", "Germany", "South Africa" عن عملاء في واحدة من الدول الثلاثة التي تم تحديدها.

ملائمة Pattern Matching: يختبر ما إذا كانت قيمة سلسلة تلائم نقش محدد. على سبيل المثال، يبحث Last Name Like "M*" عن موظفين يبدأ اسمهم الأخير بالحرف M.

اختبار قيمة Null Value Test .Null: يختبر ما إذا كانت قيمة لها قيمة Null. على سبيل المثال، يبحث Required Date Is Null عن ترتيبات دون تاريخ مطلوب.

يمكن إنشاء أبحاث معقدة بدمج أكثر من أربعين شرط للبحث باستخدام عمليات منطقية مثل AND أو OR.

بعد استخدام تقنية من تقنيات متعددة لبدء البحث، يستعيد محرك قاعدة البيانات السجلات التي تفي في شروط البحث. تتم الإشارة غالبا إلى شرط البحث SQL a WHERE clause without the WHERE"

فتح نموذج أو تقرير بسجلات تم تحديدها

لتحديد سجلات عند فتح نموذج أو تقرير لأول مرة، يمكنك القيام بأي مما يلي:

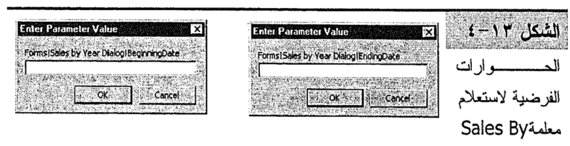
- استخدام استعلامات معلمة كمصدر سجل النموذج أو التقرير.
- عين خاصية مثل Recordsource أو Filter في إجراء يعمل عندما يتم فتح النموذج أو التقرير.
- استخدام واحدة من طرق VBA لتطبيق شرط البحث لمصدر السجل الموجود عندما يفتح النموذج أو التقرير.

استخدام استعلامات معلمة

استعلام المعلمة هو استعلام يتطلب معلومات إضافية قبل إن يشتغل. عندما تستخدم استعلام معلمة كمصدر سجل لنموذج أو تقرير، يمكنك استخدام تقنين لدعم المعلومات الإضافية للاستعلام:

- يمكنك استخدام مربع الحوار الفرضي الذي يعرضه Access تلقائيا عندما يحتوي معيار الاستعلام على معلمة.
- يمكنك استخدام تحكمات في نموذج آخر لجمع المعلومات عندما يحصل الاستعلام على المعلومات عن طريق النموذج، فأنت تستخدم تقنية Query By Form.

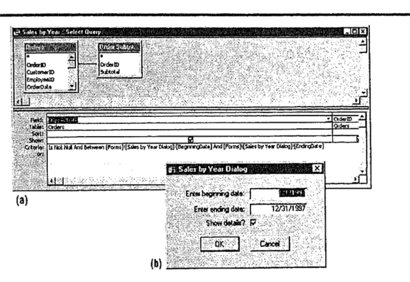
كمثال، تقرير Sales By Year له استعلام معلمة Sales By Year كمصدر سجل خاص به. إذا قمت بتشغيل استعلام Sales By Year عن طريق تحديده في إطار Data base ونقر زر Open، يعرض Access الحوارين الفرضيين للمعلمات في الاستعلام "انظر شكل ١٣-٤".



الشكل ١٣-٤ الحوارات الفرضية لاستعلام معلمة Sales By Year.

يوضح الشكل ١٣-٥ أ المعلمتين المضمنتين في تعبير المعيار لحقل Shipped Data بناء الجملة المعلمة لرمز مثل Form! [Sales By Year Dialog]! Beginning Data يوضح إن استعلام المعلمة مصمم للحصول على القيم الخاصة به من نموذج Sales By Year Dialog مع ذلك، عندما لا يكون النموذج مفتوحا، يعرض Access الحوارات الفرضية لجمع المعلومات الإضافية.

إذا قمت بتشغيل استعلام Sales By Year بفتح تقرير Sales By Year يعرض Access نموذج Sales By Year Dialog "انظر الشكل ١٣-٥ ب". بعد إن تقوم بتحديد التواريخ وإغلاق النموذج، يعمل استعلام Sales By Year مستخدما المعلومات التي أدخلتها في النموذج Query By Form، ثم يفتح تقرير Sales By Yearويعرض السجلات التي تم تحديدها.



الشكل ١٣-٥ استعلام معلمة "أ" ونموذج حوار التخصيص "ب" الذي يدعم القيم الخاصة بالمعلومات.

تعيين خاصية RecordSource

يمكن تعيين خاصية Recordsource الخاصة بالنموذج أو التقرير عن طريق كتابة إجراء حدث يتم إطلاقه بواسطة حدث Open يمكنك تعيين خاصية Recordsource لاسم الاستعلام المخزن أو عبارة SQL، مع ذلك، عند تعيين الخاصية في إجراء VBA يجب استخدام تعبير سلسلة.

على سبيل المثال، لعرض سجلات عملاء الأرجنتين في تقرير Customer Labels، افتح تقرير Customer Labels في أسلوب عرض Design انقر خاصية حدث Open الخاصة بالتقرير وانقر زر الخاصة بالتقرير وانقر زر Build في يمين مربع الخاصية. قم بإنشاء إجراءات الحدث التالي تحدد عبارة SQL سجلات لعملاء الأرجنتين من جدول Customer:

```
Private Sub Report_Open"Cancel As Integer"
   Me.RecordSource = "SELECT * FROM Customers WHERE Country = "
_
   & "'Argentina'"
End Sub
```

احفظ التقرير وبدل Print Preview. يتم إطلاق حدث Open ويعرض التقرير السجلات التي تم تحديدها. بدل لأسلوب عرض Design.

تعيين خاصية Filter

للنموذج أو التقرير خاصية Filter يمكنك استخدامها لتحديد السجلات وخاصية Filter on التي تستخدمها لفتح أو إغلاق التصفية وخاصية Filter هي تعبير سلسلة صالح كجملة SQL WHERE دون كلمة <WHERE يمكنك تعيين خاصية Filter لتحديد السجلات في إجراء حدث تم إطلاقه بواسطة حدث Open الخاص بالنموذج أو التقرير.

كمثال، قم بتعديل إجراء حدث لحدث Open الخاص بتقرير Customer Labels كما يلي:

```
Private Sub Report_Open"Cancel As Integer"
   Me.Filter = "Country = 'UK'"
   Me.FilterOn = True
End Sub
```

احفظ التقرير وبدل إلى Print Preview يتم عرض السجلات التي تحديدها.

استخدام طريقة VBA

هناك وسيطتان لطريقتي OpenForm وOpenReport لكائن Docmd، وهذه الوسيطتان يمكن استخدامهما لتحديد السجلات وهما وسيطتا Wherecondition وfiltername وسيطة Wherecondition هي تعبير سلسلة وهو جملة SQL WHERE صالحة بدون كلمة WHERE، على سبيل المثال، لفتح نموذج Customer لعرض عملاء الأرجنتين، استخدم العبارة التالية:

```
DoCmd.OpenForm formname:="Customers", wherecondition:= "Country
= " _   & "'Argentina'"
```

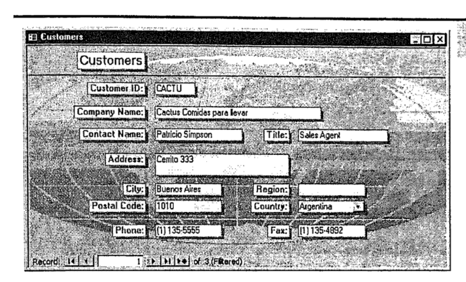
وسيطة Filtername هي تعبير سلسلة هو اسم لاستعلام تصفية تم تخزينه في قاعدة بيانات. عليك أولا إنشاء استعلام تصفية يقوم بتصفية جدول النموذج أو استعلام أو التقرير الذي تم تضمينه حتى تحدد السجلات.

يجب إن يتضمن استعلام التصفية كل الحقول في مجموعة تصفييه النموذج أو التقرير، كما يمكن إن يكون استعلام التصفية استعلام معلمة أيضا.

لطريقة ApplyFilter لكائن Docmd وسيطة Wherecondition ووسيطة Filtername. يمكنك تشغيل طريقة ApplyFilter تلقائيا عندما يفتح النموذج أو التقرير عن طريق إنشاء إجراءات حدث لحدث Open. على سبيل المثال، لتحديد عملاء الأرجنتين، يمكنك استخدام الإجراء التالي لنموذج Customers:

```
Private Sub Form_Open"Cancel As Integer"
   DoCmd.ApplyFilter wherecondition:= "Country = 'Argentina'"
End Sub
```

يوضح شكل ١٣-٦ نتيجة فتح نموذج Customers وإطلاق هذا الإجراء.



الشكل ١٣-٦ يمكنك استخدام طريقة ApplyFilter في إجراء تم إطلاقه عن طريق حدث Open الخاص بالنموذج لتصفية السجلات عند فتح النموذج.

تغيير التحديد في نموذج أو تقرير مفتوح

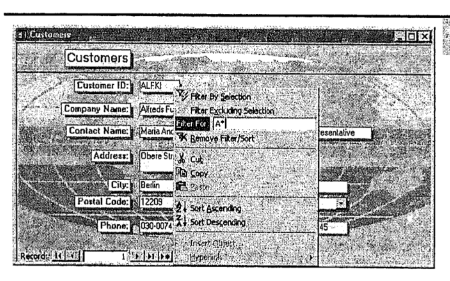
بعد فتح النموذج أو التقرير وعرض السجلات، يمكنك استخدام تقنيات عديدة لتغيير السجلات المعروضة:

- استخدام أمر Filter For المضمن والذي يتيح للمستخدم إدخال معيار تصفية في مربع نص في قائمة الاختصار.
- استخدام إطار Filter By Selection وإطار Filter By Form والذين يتيحان للمستخدم إنشاء تصفية لنموذج مفتوح.
- غير خاصية RecordSource لنموذج مفتوح باستخدام إجراء
- قم بتشغيل إجراء ApplyFilter أو طريقة لتطبيق تصفية لنموذج مفتوح.
- غير خاصية Filter لنموذج مفتوح ثم شغل التصفية في إجراء.

من كل التقنيات المذكورة، لا يمكن استخدام سوى التقنية التي تقوم بتعيين خصائص Filter وfilter On لتحديد السجلات لتقرير مفتوح، أما التقنيات الأخرى فهي تطبق فقط على تحديد السجلات في النماذج المفتوحة.

استخدام Filter For و Filter By Selection و Filter By Form

يمكنك استخدام تقنيات التصفية الثلاثة المضمن والتي يوفرها Access للسماح للمستخدم بتحديد وتغيير السجلات المعروضة في نموذج مفتوح وهي Filter For و filter By Selection و filter By Form يقدم التطبيق الخاص بك هذه التقنيات إذا كان محتوياً على أوامر القائمة المضمنة أو أزرار شريط الأدوات. على سبيل المثال، لاستخدام تقنية Filter For لتحديد سجلات لنموذج Customer، قم بتطبيق أمر Filter For في التطبيق الخاص بك حيث يستطيع المستخدم النقر يميناً في تحكم وطبع المعيار في مربع نص Filter For "انظر شكل ١٣ ٧" وضغط Enter لتطبيق التصفية.



الشكل ١٣-٧ انقر يميناً في تحكم لعرض أمر Filter For المضمن.



لاستخدام تقنية Filter For لتحديد السجلات لنموذج Customer Order، ضمن أمر Filter By Form أو زر شريط الأدوات Filter By Form يوضح شكل "١٣-٨" التصفية للبحث عن الترتيبات الموجودة بعد ١٩٩٧/١/١ بواسطة عملاء الأرجنتين. ونقر زر Filter في شريط الأدوات يطبق التصفية.

يمكنك أيضاً تخصيص واجهة التطبيق Filter By Form يمكنك إنشاء إجراءات حدث لأحداث Filter و apply Filter لتخصيص العملية.

الشكل ١٣-٨ استخدام إطار تصفية Filter By From لإنشاء تصفية.

Record source: يتعرف النموذج على حدث Filter عندما تنقر زر Filter By Form أو تختار Records ⇦ Filter ⇦ Filter By From or Advanced Filter/Sort. أو "لا يتم التعرف على حدث Filter عندما تستخدم تقنية Filter By Selection". يقوم Access بتشغيل إجراء الحدث قبل عرض واجهة تطبيق Filter By From أو استبدالها بإطار تصفية التخصيص الخاص ب.كبعد عمل إجراءات الحدث، يتم عرض واجهة التطبيق المخصصة Filter By From.

كمثال، إذا لم تكن تريد عرض النموذج الفرعي الثاني في نموذج Customer Order عندما تكون واجهة تطبيق Filter By Form نشيطة، يمكنك استخدام إجراء الحدث التالي لإخفاء النموذج الفرعي:

```
Private Sub Form_Filter"Cancel As Integer, FilterType As Integer"
    Me.[Customer Orders Subform2].Visible = False
End Sub
```

يستخدم Access وسيطة Filter Type لإخبار إجراء الحدث أيا من إطاري التصفية يحاول المستخدم فتحه، يستخدم Access الثوابت الحقيقية ac Filter Advanced وac Filter By From يمكنك تضمين عبارات تحدد قيمة وسيطة Filter Type وتضمن إرشادات تعتمد على إطار التصفية الذي يحاول المستخدم فتحه.

استخدام حدث Apply Filter

يتعرف النموذج على حدث Apply Filter.عندما يحدد النموذج واحد من ثلاثة أنواع للإجراءات وهي إزالة التصفية أو تطبيق التصفية إغلاق إطار التصفية يتعرف النموذج على حدث Apply

Filter. سواء بالتجاوب مع التفاعلات مع المستخدم أو بالتجاوب مع الإجراءات. يحـدث حـدث Apply Filter.عندما ينقر المستخدم على زر Apply Filter. أو زرRemove Filter في شريط الأدوات أو عندما يختار Records ⇦ Filter/Sort or Remove Filter/Sort. أو عندما ينقـر زر Filter By Selection في شريط الأدوات أو عندما يختار Records ⇦ Filter ⇦ Filter By Selection. يحدث حدث Apply Filter.أيضا عندما يشغل الإجراء إجراءاً أو طريقة تطبـق أو تزيل تصفية مثل إجـــراءات أو طـرق Apply Filter. أو Open Form أو Show All Record أو عندما يغلق إطار التصفية أو يعين خاصية Filter أوFilter on يمكنك إجراء حـدث للقيام بعمليات مختلفة معتمدا على أي الإجراءات الثالثة قم بإطلاق الحدث. بناء جملة إجـــراءات حدث Apply Filter.هي:

Form_ApplyFilter "Cancel As Integer, ApplyType As Integer

يمكنك استخدام وسيطة Cancel لحذف السلوك الفرضي الذي يتبع الحدث "مطبقا التصفيـة أو مزيلا للتصفية أو مغلقا لإطار التصفية". يستخدم Access وسيطة Apply Type لإخبار إجـــراء الحدث بالإجراء الذي تسبب في الحدث ويقوم Access بتعيين وسيطة Apply Type لواحد مـــن الثوابـــت الحقيقيـة acShowAllRecords, أو acApplyFilter, أو acCloseFilterWindow يمكن أن يحتوي إجراء الحدث على مجموعات مختلفة من العبارات لكل بديل. يشتغل الإجـــراء قبل تطبيق التصفية فعليا أو قبل إزالة التصفية أو بعد إغلاق إطار التصفية لكـــن قبـل عـــرض النموذج مرة أخرى. بالتالي يمكنك استخدام عبارات الإجراء للقيام بما يلي:

- تعديل التصفية قبل تطبيقها.على سبيل المثال، يمكنك قراءة وتغيير خاصية Filter.
- تغيير كيفيه عرض النموذج عند التصفية. على سبيل المثال، إذا كنت تقـــوم بالتصفيـــة لتحديد الترتيبات المدفوعة، يمكنك إخفاء تحكمات في نموذج Order والتي تكون غـــير ملائمة للتصفية مثل "Amount Due".
- إلغاء أو تغيير الإجراءات التي اتخذها الإجراء عندما حدث Filter. على سبيل المثـــال، إذا أخفيت تحكمات في واجهة تطبيق Filter By From عندمـــا حـــدث Filter يمكنـــك عرض هذه التحكمات بعد تطبيق التصفية.

يمكنك إعادة عرض النموذج الفرعي الثاني في نموذج Customer Order كما يلي:

```
Private Sub Form_ApplyFilter"Cancel As Integer, ApplyType As Integer"
    Me.[Customer Orders Subform2].Visible = True
End Sub
```

تغيير خاصية Records

بعد فتح النموذج، يمكن تغيير خاصية النموذج RecordsourceلعبارةSQL أو استعلام مخزن يحدد السجلات. كمثال، افتح نموذج Customers في أسلوب عرض Form وأدخل العبارة التالية في إطار Immediate:

```
Forms!Customers.RecordSource = "SELECT * FROM Customers WHERE
" _    & "Country = 'Argentina'"
```

يتم تغيير مصدر السجل وعرض السجلات المحددة.

ملاحظة لا يمكنك تغيير خاصية Recordsource الخاصة بالتقرير بعد فتح التقرير.

استخدام طريقة Apply Filter

طريقة أخرى لتغيير السجلات المعروضة في نموذج مفتوح هي استخدام طريقة Apply Filter لكائن Docmd. يمكنك استخدام وسيطة wherecondition لتحديد السجلات باستخدام تعبير سلسلة يكون جملة SQLWHERE صالح بدون كلمة WHERE. يمكن أن يكون استعلام معلمة يحصل على المعلومات الخاصة به من التحكمات في نفس النموذج "مل مربع تحرير وسرد لتحديد موضوع في عنوان النموذج" أو تحكمات نموذج في نموذج آخر مثل حوار تخصيص مصمم لتجميع معايير تحديد. يتضمن القسم التالي مثالاً لاستخدام استعلام معلمة كاستعلام تصفية.

تغيير خاصية Filter

للنموذج أو التقرير خاصية Filter يمكنك استخدامها لتحديد السجلات وخاصية Filter on تستخدمها لفتح أو إغلاق التصفية. خاصية Filter هي تعبير سلسلة يكون جملة Sqlwhere صالحة بدون كلمة Where. بعد فتح النموذج أو التقرير، يمكنك تعيين خاصية Filter الخاصة بالنموذج ثم تعيين خاصية Filter on إلى True

لاكتشاف خاصية Filter، اتبع الخطوات التالية:

١- افتح تقرير Product By Category في أسلوب عرض تقرير Design لاحظ أن خاصية Filter وخاصية Filter on تكون NO. بدل إلى Print Preview

٢- اطبع Reports![Products by Category].Filter = "CategoryName = 'Beverages'" في إطار Immediate واضغط Enter. لا تغير Print Preview للتقرير لأن خاصية Filter On هي No بدل لأسلوب عرض Design لتلاحظ أن

خاصية Filter ق تغيرت إلى Category = 'Beverages'. بدل مرة أخرى إلى Print Preview.

٣- اطبع Reports![Products by Category].FilterOn = True في إطار Immediate واضغط Enter يتغير إطار التقرير لعرض السجلات التي تمت تصفيتها. بدل لأسلوب عرض Design لتلاحظ أن خاصية Filter On قد تغيرت إلى Yes.

٤- بدل مرة أخرى إلى Print Preview. اطبع Reports![Products by Category].Filter = "CategoryName = 'Condiments'" في إطار Immediate واضغط Enter. يتغير إطار التقرير لعرض السجلات المصفاة. وبمجرد كون خاصية Filter On في Yes، يمكنك تغيير التصفية بتعيين خاصية Filter.

٥- اغلق التقرير، إذا أغلقت التقرير دون حفظه، يتجاهل Access تعيينات Filter وfilter On إذا أغلقت التقرير بعد حفظه، يتجاهل Access تعيين Filter لكن يحفظ تعيين Filter On.

البحث عن مجموعة من السجلات باستخدام Query By Form

عملية قادة بيانات أساسية هي تحديد مجموعة من السجلات تتف مع معيار تحديد واد أو اكثر. على سبيل المثال، في نموذج Order Status يمكن أن تريد عرض قائمة بأوامر عميل ما أو موظف ويمكن أن تريد مراجعة كل الأوامر التي صدرت لموظف بعد تاريخ محدد أو رؤ ية كل الأوامر التي ستشحن بواسطة شاحن ما قبل تاريخ محدد. عند العمل تبادليا، تحدد مجموعة من السجلات عن طريق إنشاء وتطبيق تصفية، يوفر Access طرقاً متعددة لإنشاء التصفيات تبادليا تتضمن Filter For وfilter By From وfilter By Selection وfilter Excluding. تعطي هذه التقنيات المستخدم إمكانات استعلامية قوية. يمكنك أيضا تخصيص الإمكانات المضمنة هذه عن طريق البرمجة كما تم التوضيح في القسم السابق. تقنية أخرى هي استخدام Query By Form والتي توفر واجهة تطبيق بسيطة لتحديد مجموعة من السجلات.

وفيما يلي تلخيص للخطوات الخاصة باستخدام تقنية Query By Form:

- حدد الحقل الذي تريد استخدامه للتحديد وتأكد من أن الحقل موجود في مصدر سجل النموذج. على سبيل المثال، لتحديد سجلات لعميل في نموذج frm Order Status، أضف حقل Customer ID من جدول Orders لمصدر سجل النموذج.
- ضع مربع تحرير وسرد غير مقيد في عنوان النموذج أو تذييله والذي يحمل قيمة البحث لحقل التحديد. صمم مربع التحرير والسرد لعرض قائمة لتحديد قيمة البحث على سبيل المثال، قم بإنشاء مربع تحرير وسرد يدعى cboCustomer بقائمة تحتوي على

Customer ID كعمود مربع التحرير والسرد المقيد والمختفـــي وcompany Name كعمود معروض "يحدد المستخدم اسم الشركة ويحمل مربع التحرير والسرد ID الخـاص بالعميل المقابل".

- قم بإنشاء استعلام تصفية يعتمد على مصدر سجل النموذج والذي يحدد سجلات تناســب قيمة البحث الموجودة في مربع التحرير والسرد. على سبيل المثال، لتحديد أوامر لعميـل في نموذج Order Status، أنشئ استعلام تصفية يدعـــى qfltOrder Status يعتمــد على مصـــدر ســجل النمـــوذج ثــم عيـــن المعيــار لحقـل Customer Id إلـــى. !frmOrderStatus!cboCustomer في حالة Filter Excluding Selection، ستقوم بإنشاء استعلام تصفية يحدد السجلات التي لا تتناسب مع قيمة البحث الموجودة في مربع التحرير والسرد.
- قم بإنشاء إجراء حدث يستخدم طريقة Apply Filter لكائن Docmd بتشـــغيل اســـتعلام التصفية وعرض السجلات التي تمت تصفيتها عندما يغير المستخدم القيمة فـــي مربـــع التحرير والسرد ويتعرف مربع التحرير والسرد على حدث AfterUpdate.

يستخدم إجراء حدث cboCustomer Afterupdate التالي، طريقة Apply Filter لكــــائن Docmd لتطبيق التصفية على نموذج Order Status.

```
Private Sub cboCustomer_AfterUpdate""
    DoCmd.ApplyFilter "qfltOrderStatus"
End Sub
```

عندما تعمل تبادليا، يمكنك إزالة التصفيـــة عـــن طريـــق اختبـــار Record ⇦ Remove Filter/Sort. يمكنك جعل إزالة التصفية تلقائية بوضع زر أمـــر فـــي رأس صفحــه النمـــوذج واستخدام إجراء حدث مثل الإجراء التالي لإزالة التصفية:

```
Private Sub cmdShowAll_Click""
    DoCmd.ShowAllRecords
    cboCustomer = Null
End Sub
```

من الاساليب الأخرى لعرض كل السجلات هي عرض صف Null في قائمة السرد والتحرير وعندما يحدد المستخدم صف Null يتم عرض كل السجلات.

استخدام معايير متعددة لتحديد مجموعة من السجلات

في اغلب الأوقات تريد استخدام اكثر م معيار تحديد واحد. على سبيل المثال، يمكنـــك اســـتخدام نموذج frm Order Status للبحث عن كل الأوامر لعميل والتي يعالجها موظفـــون محـــددون

يمكنك وضع مربع سرد وتحرير للتحديد في رأس صفحة النموذج لكل الحقل تريد استخدامه لتحديد السجلات وتهيئ التطبيق لأي من الطريقتين الآتيتين:

- ♦ عدل استعلام التصفية ليتضمن المعيار الإضافي لكل مربع تحرير وسرد.
- ♦ قم بإنشاء إجراء لكل مربع سرد وتحديد يطبق التصفية
- ♦ عدل الإجراء الذي يزيل التصفية بحيث يعين الإجراء كل مربعات التحرير والسرد إلى Null "انظر شكل ١٣-٩".

Order Status

Show All | Select Customer: Alfreds Futterkiste | Select Employee:

Order ID	Order Date	Required Date	Shipped Date	Shipper	Customer	Employee
10835	15-Feb-96	14-Mar-96	21-Feb-96	Federal Shipping	Alfreds Futterkiste	Davolio
10952	15-Apr-96	27-May-96	23-Apr-96	Speedy Express	Alfreds Futterkiste	Davolio
11011	09-May-96	06-Jun-96	13-May-96	Speedy Express	Alfreds Futterkiste	Leverling
10692	03-Nov-95	01-Dec-95	13-Nov-95	United Package	Alfreds Futterkiste	Peacock
10702	13-Nov-95	25-Dec-95	21-Nov-95	Speedy Express	Alfreds Futterkiste	Peacock
10643	25-Sep-95	23-Oct-95	03-Oct-95	Speedy Express	Alfreds Futterkiste	Suyama

Record: 1 of 6 (Filtered)

الشكل ١٣-٩ مربعات التحرير والسرد لتحديد عميل وموظف وزر الأمر لإزالة التصفية.

في المثال الخاص بنا، سنحتاج لإضافة حقل Employee ID من جدول Order لمصدر سجل النموذج قبل أن تبحث بواسطة الموظف. وفيما يلي إجراء الحدث لمربع التحرير والسرد:

```
Private Sub cboEmployee_AfterUpdate""
   DoCmd.ApplyFiler "qfltOrderStatus"
End Sub
```

وفيما يلي نوضح إجراء الحدث لزر الأوامر الذي يقوم بإزالة التصفية.

```
Private Sub cmdShowAll_Click""
   DoCmd.ShowAllRecords
   cboCustomer = Null
   cboEmployee = Null
End Sub
```

تزامن مربعين تحرير وسرد

عندما تكون القائمة المعروضة في مربع التحرير والسرد طويلة، يمكنك استخدام مربعين تحرير وسرد بدلا من مربع واحد وتصميمهما بحيث تعتمد المحتويات المعروضة من قبل مربع التحرير والسرد الثاني على القيمة التي تحددها في المربع الأول، بمعنى أن مربع التحرير الثاني متزامن

مع الأول. على سبيل المثال، في قاعدة بينات ترتيبات، تحتوي العملاء على مئات الأسماء، لذلك يمكنك استخدام مربع تحرير وسرد لتحديد الحرف الأول أو الحرفين واستخدام مربع تحرير وسرد ثاني لعرض أسماء الشركات الخاصة بكل العملاء الذين تبدأ أسمائهم بالحرف أو الحرفين.

يبحث إجراء حدث cboFind AfterUpdate لمربع تحرير وسرد CboFind عن عميل:

```
Private Sub cboFind_AfterUpdate
   Me.RecordsetClone.FindFirst "CustomerID = """ & Me!cboFind & """"
   Me.Bookmark = Me.Recordsetclone.Bookmark
End Sub
```

لأن Customer ID له قيم سلسلة يحتوي تعبير السلسلة للوسيطة الخاصة بطريقة Find First على سلسلة داخلية "" & Me!Find & """يستخدم الإجراء زوجين كمن علامات التنصيص للتعرف على السلسلة الداخلية داخل السلسلة الخارجية يشتغل إجراء حدث cboFirst_AfterUpdate لمربع تحرير وسرد cboFirst بعد أن تحدد قيمة مختلفة في مربع التحرير والسردcboFirst يتزامن مربع التحرير والسرد هذا مع مربع التحرير cboFind عن طريق إعادة تشغيل الاستعلام الخاص به ثم تحريك البؤرة لمربع التحرير والسرد cboFind:

```
Private Sub cboFirst_AfterUpdate""
   cboFind.Requery
   cboFind.SetFocus
End Sub
```

يستخدم هذا الإجراء طرقا لمربع التحرير والسرد بدلاً من استخدام طرق كائن DoCmd.

تلميح طريقة Requery لكائن أو نموذج هي أسرع من طريقة Requery لكائن DoCmd. عندما تشغل طريقة Requery DoCmd، يغلق Access الاستعلام ثم يعيد تحميل الاستعلام من قاعدة البيانات. لكن، عندما تستخدم طريقة Requery لكائن أو نموذج، يشغل Access الاستعلام دون إعادة تحميله.

لاختبار إجراء الحدث، اتبع الخطوات التالية:

١- ضع مربع تحرير وسرد غير منظم يدعى cobFind في رأس صفحة نموذج Customer. احذف بطاقة عنونة مربع التحرير والسرد. انقر في خاصية حدث مربع التحرير والسرد AfterUpdate وانقر زر Build إلى يمين مربع الخاصية. ادخل إجراءات cboFind_AfterUpdate

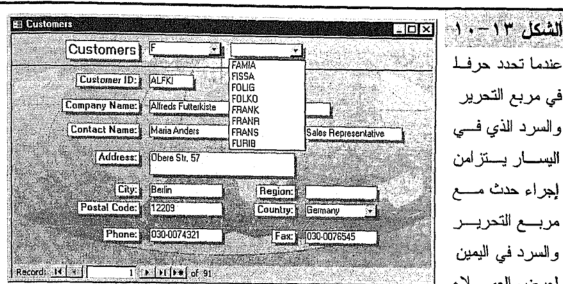
٢- ضع مربع تحرير وسرد غير منظم يدعى cobFind في قسم رأس صفحة النموذج. غير خاصية RowSource Type إلى Value Lest في خاصية RowSource. اطبع A,B,C,E,F, وما إلى ذلك إلى آخر الأبجدية، تقوم الفاصلة المنقوطة "," في بداية تعيين خاصية RowSource بإنشاء صف Null. احفظ واغلق النموذج

٣- قم بإنشاء استعلام جديد يدعى qrySecond معتمدا على جدول Customers. اسحب حقول Customers ID وcompany Name للوح الأزرار في إطار Query. في حقل Criteria الخاص بالعمود الذي يحتوي علىCustomers ID، اطبع: Like "*" & [Forms]![Customers]![cboFirst]. يبحث هذا عن كل السجلات في جدول Customers مع وجود الحرف الأول المتشابه كالذي حددته في مربع التحرير والسرد cobFirst. احفظ الاستعلام.

٤- افتح نموذج Customers في أسلوب عرض Design وحدد مربع التحرير والسرد cobFind. لخاصية Resource حدد استعلام qrySecond.

٥- انقر في خاصية حدث AfterUpdate الخاصة بمربع التحرير والسرد cobFind.وانقر زر Build في يمين مربع الخاصية. ادخل إجراء الحدث cboFirst_AfterUpdate.

٦- احفظ النموذج وبدل لأسلوب عرض Design. اختبر مربعات التحرير والسرد الخاصة بالنموذج بتحديد حرف من حقل cobFirst.ونقر القائمة المنسدلة من مربع التحرير والسرد cobFind."انظر الشكل ١٣ــ١٠". أيضا اختبر ما سيحدث عندما تستخدم صف Null في مربع التحرير والسرد cobFind.



الشكل ١٣-١٠ عندما تحدد حرفا في مربع التحرير والسرد الذي في اليسار يتزامن إجراء حدث مع مربع التحرير والسرد في اليمين لعرض العملاء الذين تبدأ أسمائهم بالحروف التي تم تحديدها.

استخدام مربع قائمة متعدد التحديد لتصفية السجلات

تعتمد تقنيات التحديد المشروحة في الأقسام السابقة هي تحيد السجلات التي تفي بظروف البحث. تستخدم تقنية مختلفة تماما في أساسها مربع القائمة متعدد التحديد لعرض قائمة بالاختبارات التي تتيح للمستخدم اختيار السجلات عشوائيا دون استخدام ظرف بحث. وهذه التقنية اكثر مرونة عن التقنيات الأخرى لأن المستخدم يستطيع اختيار السجلات الفردية للعرض.

يتيح مربع القائمة القياسي للمستخدمين تحديد عنصر واحد من القائمة. لكن، يمكنك استخدام خاصية MultiSelect للسماح للمستخدمين باختيار اكثر من عنصر لخاصية MultiSelect القيم التالية:

None: تتيح للمستخدم تحديد عنصر واحد.

Simple: تتيح للمستخدم عناصر كثيرة باستخدام كل عنصر على حدة.

Extended: يوسع تحديد عناصر متعددة. استخدم Click+Shift أو Shift + سهم لتوسيع التحديد من العنصر الذي تم تحديده سابقا للعنصر الحالي.

ملاحظة تنقر لتحديد عنصر وتضغط Ctrl+Click لعدم تحديد عنصر في مربع القائمة.

لاكتشاف مربع القائمة متعدد التحديد، سنقوم بإنشاء نموذج يعرض الترتيبات في مربع القائمة متعدد التحديد. سنستخدم مربع تحرير وسرد لتحديد موظف وجعل مربع القائمة متزامناً مع مربع التحرير والسرد باستخدام إجراء حدث تم إطلاقه بواسطة حدث AfterUpdate الخاص بمربع التحرير والسرد. يعرض إجراء الحدث هذا والموضح فيما يلي، الترتيبات في مربع القائمة الذي يقابل القيمة التي تم تحديدها في مربع التحرير والسرد.

```
Private Sub cboCurrent_AfterUpdate
   lstOrders.Requery
End Sub
```

اتبع هذه الخطوات لإعداد النموذج:

١- قم بإنشاء نموذج غير منظم يدعى frmAssignOrder عين خصائص النموذج التالية:

Caption	Assign Orders
ScrollBars	Neither
RecordSelectors	No
NavigationButtons	No
MinMaxButtons	None

٢- ضع مربع تحرير غير منظم يدعى cboCurrent بالخصائص التالية:

RowSource	SELECT EmployeeID, LastName & ", " & FirstName As FullName FROM Employees ORDERBY LastName;
ColumnCount	2
BoundColumn	1
ColumnWidths	0";1"

٣- عين خاصية Caption لبطاقة عنونة مربع التحرير والسرد إلى Current Sales Person.

٤- قم بإنشاء استعلام المعلمة الذي يدعي qrySortCountry. اسحب حقول Order ID وorder Date وEmployee ID من جدول Orders وحقول Country وcompany Name من جدول Customers للوح الأسفل لإطار Query. في حقل Criteria

للعمــود الـــذي يحتـــوي علــــى حقــــل Employee ID، الطبــــع [Forms]![fmAssignOrder]![cboCurrent] "انظر شـكل ١٣-١١". قــم بفــرز حقل Country وcompany Name في ترتيب تصاعدي. يستخدم هذا الاستعلام قيمة Employee ID التي تم اختيارها في مربع التحريـــر والســرد cboCurrent لتحديــد السجلات "Query By From" سنستخدم هذا الاستعلام كمصدر الصف لمربع القائمـــة متعدد التحديد.

الشكل ١٣-١١ يحصـــل استعـــلام المعلمـــة لمربـــع القائمة على المعيار الخاص بـــه مـــن مربـــع التحريـــر والســـرد فـــي النموذج.

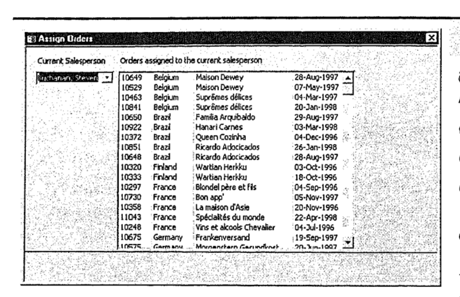
٥- قم بإنشاء مربع قائمة غير منظم يدعى 1stOrder بالخصائص التالي

RowSource	QrySortCountry
ColumnCount	4
ColumnWidths	0.5; 0.65; 1.5; 0.5
BoundColumn	1
MultiSelect	Simple

٦- عين خاصية Caption لبطاقة عنونة مربـــع القائمـــة الـــي Orders assigned to .current sales person

٧- ادخل إجراء الحدث CboCurrent_AfterUpdate لخاصيـــة حـــدث AfterUpdate الخاص بمربع التحرير والسرد. يتطلب إجراء الحدث مربع القائمة عندما تختار موظـف مختلف في مربع التحرير والسرد CboCurrent.

٨- احفظ النموذج وبدل لأسلوب عرض Form.حدد موظفا في مربـع التحريـــر والســـرد current sales person.يوضح شكل ١٣_١٢النموذج بعـــرض ترتيبـــات Steven :Buchanan



الشكل ١٣-١٢

يعرض نموذج Assign Order غير المنظم سجلات من جدول Employees كمصدر صف لمربع التحرير والسرد وسجلات من استعلام تعتمد على جداول Order و customers كمصدر صف لمربع القائمة متعدد التحديد. يحافظ إجراء الحدث على مربع القائمة متزامناً مع مربع التحديد والسرد.

فرز صفوف مربع القائمة عن طريق RowSource

يمكنك جعل مربع القائمة اكثر فائدة عن طريق السماح للمستخدم بفرز الصفوف عن طريق أي من الحقول عن طريق تغيير ترتيب الفرز للصفوف في مربع القائمة أو مربع التحرير والسرد هي تغيير خاصية RowSource. سنضع مجموعة من أربعة أزرار تحكم في اسفل مربع القائمة ونستخدم كل زر لتغيير خاصية RowSource الخاصة بمربع القائمة لاستعلام مختلف. على سبيل المثال، فيما يلي إجراء حدث لزر cmdSortOrderID:

```
Private Sub cmdSortOrderID_Click
   Me!lstOrders.RowSource = "qrySortOrderID"
End Sub
```

١- قم بإنشاء استعلامات

كنسخ من qrySortOrderDate و qrySortCompany و qrySortOrderID و qrySortCountry

لكن بترتيبات الفرز التالية:

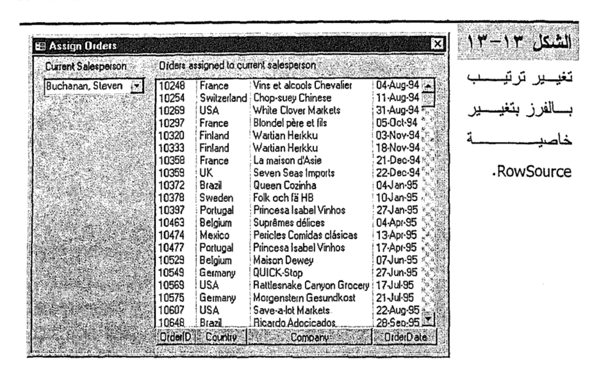
Sort Query	OrderID	CompanyName	OrderDate
QrySortOrderID	Ascending		
QrySortCompany	Ascending		
QrySortOrderDate		Ascending	

٢- ضع مجموعة من أربعة أزرار للأمر تدعى cmdSortCountry cmdSortID و cmdSortCompany و cmdSortOrderDate بامتداد اسفل حافة مربع القائمة بالعناوين OrderID و country و company و orderDate على التوالي.

٣- قم بإنشاء إجراء حدث لكل زر أمر يعين خاصية RowSource لمربع الحوار للاستعلام المقابل ، انظر إجراء cmdSortOrderID _Click الموضح سابقا كمثال"

٤ – احفظ النموذج وبدل لأسلوب عرض Form. حدد Steven Buchanan واختبر أزرار الفرز" انظر شكل ١٣-١٣".

ملاحظة إذا نقرت واحداً من أزرار أمر محدداً مصدر الصف الأربعة قبل تحديد Current SalesParson من مربع التحرير والسرد، سيفتح مربع القائمة السجلات بالفرز الذي حددته. هذا لأنه قد عين بالفعل خاصية RowSource لمربع القائمة حتى إذا لم يكن مربع القائمة مطلوب بالفعل.

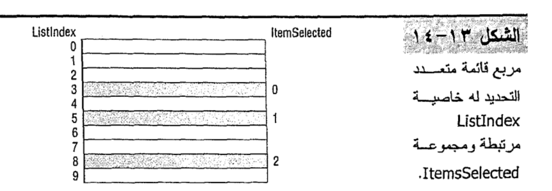


الشكل ١٣-١٣ تغيير ترتيب بالفرز بتغيير خاصية .RowSource

الحصول على معلومات مربع قائمة متعدد التحديد

عندما تحدد عناصر متعددة في قائمة ما، يقوم Access بإنشاء مجموعة لمربع القائمة تدعى مجموعة ItemSelected.كل عضو في مجموعة ItemSelected هو عدد صحيح يشير إلى صف محدد في مربع القائمة أو مربع التحرير والسرد. حتى بالرغم من كون أعضاء مجموعة ItemSelected إعداد صحيحة، نوع البيانات الخاص بهم Variant فرضيا.

عندما تعمل مع مربع قائمة متعدد التحديد، يمكن أن يكون لديك رقمي فهرس وهما رقم فهرس لموقعه في مربع القائمة ورقم فهرس آخر "أو رقم عنصر" لموقعه في مجموعة ItemSelected كما هو مبين في شكل ١٣ ١٤.ورقما الفهرس كلاهما يعتمدان على الصفر. يمكنك استخدام خاصية LestIndex لمربع القائمة لتحديد موقع صف لمربع في مربع القائمة. عندما تحدد صفا في مربع قائمة متعدد التحديد، يعين Access تلقائياً الصف في رقم العنصر الخاص به. وحيث أن مجموعة ItemSelected تعتمد على الصفر هي الأخرى،تتم الإشارة لأول عنصر تم تحديده "ItemSelected"0 وتتم الإشارة لثاني عنصر تم تحديده "ItemSelected"1 وما إلى ذلك إذا كان العنصر الأول الذي تم تحديده في الصف الرابع من القائمة "بموقع رقم ٣"،إذن "ItemSelected"0 يرجع إلى ٣.



الشكل ١٣-١٤ مربع قائمة متعـــدد التحديد له خاصيـــة ListIndex مرتبطة ومجموعـــة ItemsSelected.

ترجع كلا من خاصيتي ListIndex ومجموعة ItemsSelected رقم موقـــع الصـــف. إذا أردت إرجاع البيانات في صف،استخدم خاصية Column أو طريقة ItemData. تستخدم طريقة ItemData لإرجاع البيانات في العمود المنظم لصف محدد عن طريق رقـــم موقعـــه تســـتخدم خاصية Column لإرجاع البيانات في عمود محدد لصف تم تحديـــده بواســـطة رقـــم موقعـــه. ولنكتشف هذه المفاهيم في إطار Immediat.

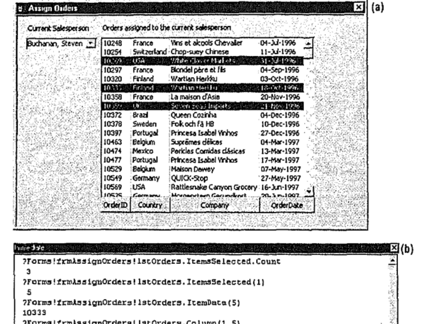
١- حدد Steven Buchanan وانقر زر الفرز OrderID. حدد صف ٣ وصف ٦ وصف ٨ "انظر شكل ١٣ــ١٥ أ".

٢- افتح لإطار Immediate، اطبع ?Forms!frmAssignOrders!lstOrders.ItemsSelected.Count, واضغط Enter يعرض إطار Immediate ٣.

٣- لعـــرض رقـــم الصـــف الـــذي يقـــابل العنصـــر الثـــاني فـــي المجموعـــة، اطبـــع ?Forms!frmAssignOrders!lstOrders.ItemsSelected"1" واضغـــــط Enter يعرض إطار Immediate ٥ "الصف ٦ له موقع ٥".

٤- لعرض البيانـــات فـــي العمـــود المنضـــم للعنصـــر الثـــاني فـــي المجموعـــة اطبـــع ?Forms!frmAssignOrders!lstOrders.ItemData"5" واضغـــط Enter إطـــار Immediate يعرض OrderID للعنصر ١٠٩٢٢.

٥- لعرض البيانات في اي عمود للعنصر الثاني في المجموعة، تستخدم خاصيـــة Column لمربع القائمة أو مربع التحرير والسرد الوسيطة الأولى لخاصية Column هـــي رقـــم العمود الذي تريد عرضه والوسيطة الثانية هي رقم الصف الوسيطتان تعتمـــدان علـــى صفـــر. أذن، لعـــرض العمـــود الثـــاني للعنصـــر الثـــاني فـــي المجموعـــة اطبـــع ?Forms!frmAssignOrders!lstOrders.Column"1,5" واضغـــط Enter إطـــار Immediate يعرض Finland "انظر شكل ١٣ــ١٥ ب".



الشكل ١٣-١٥ ثلاثة صفوف تم تحديدها في مربع التحديد "أ". استخدم مجموعة ItemSelected لتعقب التحديدات في مربع القائمة واستخدم طريقة ItemData أو Column خاصية مربع القائمة لعرض البيانات "ب".

تجهيز مربع القائمة متعدد التحديد للعمل

والآن بعد وجود طريقة لتحديد العناصر التحكمية في مربع القائمة، يمكننا تشغيل مربع القائمة. افترض أن موظفاً جديداً تم تعيينه وتريد إعادة تعيين بعض الأوامر التي يلزم بها حاليا موظفون آخرون للموظف الجديد. يمكنك وضع مربع تحرير وسرد ثاني في النموذج لعرض اسم الموظف الذي تريد إلزامه بالأوامر واستخدام زر أمر للقيام بإعادة التعيينات.بعد ان تحدد الأوامر الذي تريد تعيينها، نقر زر الأمر يشغل إجراء حدث ينتقل خلال مجموعة الأوامر التي تم تحديدها. وإجراء الحدث هو كما يلي:

```
Private Sub cmdAssign_Click
   Dim db As Database, rst As Recordset, msg As String
   Dim varNumber As Variant
   If IsNull"cboCurrent" Or IsNull"cboNew" Then
      msg = "You must select a current employee and another"
      msg = msg & " employee you want to reassign orders to."
      MsgBox msg
      Exit Sub
   End If
   If lstOrders.ItemsSelected.Count = 0 Then
```

```
      MsgBox "You must select at least one order to reassign."
      Exit Sub
   End If
   Set db = CurrentDB
   Set rst = db.OpenRecordset""Orders", dbOpenTable"
   rst.Index = "PrimaryKey"
   ' Set up the For Each loop through the collection
   For Each varNumber In lstOrders.ItemsSelected
      rst.Seek "=", lstOrders.ItemData"varNumber"
      rst.Edit
      rst!EmployeeID = cboNew
      rst.Update
   Next
   cboCurrent = cboNew
   cboNew = Null
   lstOrders.Requery
End Sub
```

لكل أمر في المجموعة، يغير الإجراء الاسم في حقل EmployeeID للموظف الجديد عندما يتم إعادة تعيين كل الأوامر، يعرض الإجراء كل الأوامر التي تم تعيينها حاليا للموظف الجديد.

يبدأ جراء بتحديد ما إذا كان مندوب المبيعات الجديد والحالي تم تحديده أم لا، ثم يعرض بعد ذلك رسالة وينهي إذا لم يكن هناك أي اختيار. بعد ذلك يحدد الإجراء ما إذا كانت أية أوامر قد تم تحديدها لإعادة التعيين ويعرض بعد ذلك رسالة وينهي إذا لم تكن هناك أوامر تم تحديدها. إذا كان هناك مندوبو تم تحديدهم وأوامر للتعيين يحدث كل هذا العمل في الذاكرة. يفتح الإجراء مجموعة سجل بنوع جدول في جدول Orders. أسرع طريقة للبحث عن الأمر الذي يقابل العنصر في المجموعة هي باستخدام طريقة Seek بتعيين الفهرس الحالي في فهرس OrderID "وهو الفهرس الأساسي الأولي .بعد إيجاد الأمر، يستخدم الإجراء طريقة Edit لنسخ السجل لمخزن النسخ ويعيد تعيين الموظف ثم يستخدم طريقة Update لحفظ التغيرات.

يستخدم الإجراء بنية For Each Next للتحليق داخل مجموعة الأوامر التي تم تحديدها ويعيد تعيين كل أمر في المجموعة بعد إعادة تعيين الأوامر، يقوم الإجراء بتعيين مربع التحرير والسردcmbCurrent يعرض لمندوب المبيعات الأوامر التي تم تعيينها له ويعيد مربع استعلام القائمة لعرض أوامر هذا الشخص

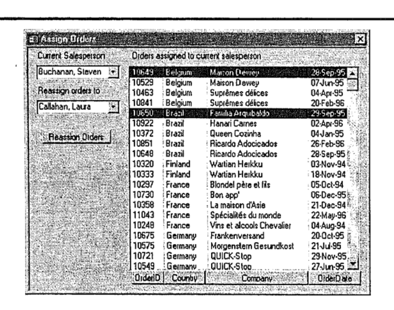
اتبع هذه الخطوات لتختبر الإجراء:

١- ضع مربع تحرير وسرد جديد يدعى CboNew في النموذج. عين خصائص RowSource وcolumn Count BoundColumn وcolumnWidths مثل مربع تحرير وسرد cboCurrent غير في بطاقة عنونة خاصية Caption إلى Reassign Orders to

٢- ضع زر أمر يدعى cmdAssign في النموذج وغير خاصية Caption إلى RieassignOrders. قم بإنشاء إجراء الحدث cmdAssign_Click الموضح في بداية هذا القسم لخاصية حدث OnClick لزر الأمر.

٣- احفظ النموذج وبدل لأسلوب عرض Form. حدد Buchanan Steven كمندوب المبيعات الحالي وlaura Callahan كمندوب المبيعات الجديد. حدد أوامر مع Order IDS من ١٠٦٤٩ و ١٠٦٥٠.انظر الشكل ١٣ـ١٦.

٤- انقر زر ReassignOrder يقوم الإجراء بإعادة تعيين الأوامر ويعرض laura Callahan كمندوب المبيعات الحالي. انقر زر OrderID وتأكد من أن الأمرين تم تعيينهما إلى laura Callahan



الشكل ١٣–١٦ إعادة تعيين الأوامر التي تم تحديدها من موظف لآخر.

استخدام تقنيات SQL

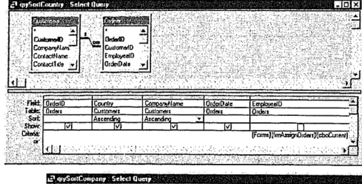
عندما تريد العمل مع مجموعة من السجلات في إجراءات VBA يمكنك استخدام مجموعتين أساسيتين من التقنيات وهما التقنيات التنقلية وتقنيات SQL. مع التقنيات التنقلية تستطيع التجول خلال مجموعات السجلات وتحديد أو تغيير سجل واحد في المرة الواحدة.على سبيل المثال، في القسم السابق، استخدمنا تقنية تنقلية للتجول خلال السجلات التي تم اختيارها في مربع قائمة متعدد التحديد وتحرير سجل في المرة الواحدة.

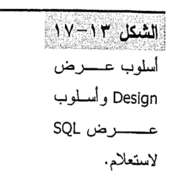
تقنيات SQL مناسبة عندما تستطيع تعريف مجموعة السجلات باستخدام استعلام أو عبارة SQL يمكنك إنشاء استعلام تحديد لتحديد السجلات أو استعلام إجراء لتعديل المجموعة وتشغيل الاستعلام في عبارة SQL واحدة. تدعى تقنيات SQL التقنيات العلاقية أو تقنيات المجموعة لأنك تعرف مجموعة من السجلات ونتيجة النهاية التي تريد من المحرك إنتاجه.لاكنك لا تحدد كيفية إتمام النتيجة.

تقنيات SQL أسرع دائما من تقنيات التنقل لأن تعتمد على الشفرة المضمنة لمحرك قاعدة البيانات. Jet وMSDE لهما التقنيات المثالية الخاصة بهما لتحديد وتعديل السجلات عندما يستخدم إجراء VBA الخاص بك تقنيات SQL لتحديد وتعديل السجلات،تتيح للمحرك طرقة الخاصة لإنتاج النتيجة التي تحددها. بطبيعة الحال، تستخدم التقنيات التنقلية فقط عندما لا يمكن تحديد مجموعة السجلات أو التغيرات التي تريد القيام بها لكل سجل باستخدام استعلام أو عبارة SQL.يتطلب استخدام تقنيات SQL إنشاء استعلامات مخزنة وكتابة عبارات SQL في Access 2000، يمكنك استعلامات بطريقتين سواء باستخدام شبكة التصميم في اسلوب عرض SQL لإنشاء استعلام SQL كعبارة SQL هو لغة قاعدة البيانات القياسية المستخدمة من قبل Access وتطبيق قاعدة البيانات الكبرى الأخرى. تستخدم SQL لكتابة عبارة تصف مجموعة البيانات التي تريد استرجاعها.

SQL مصمم لأن يكون مستقلا عن البرنامج، بمعنى أنة لا يهم ما هو تطبيق البرنامج الذي تستعمله فعليا لاسترجاع البيانات عمليا، هناك لهجات متعددة من SQL لكنها ليست تبادلية "يستخدم هذا الكتاب Access SQL للإشارة للهجة المستخدمة في أكسس".

يمكنك استخدام أسلوب عرض SQL أو أسلوب Design لإنشاء وتحرير معظم أنواع الاستعلامات. عندما تقوم بإنشاء استعلام في شبكة التصميم، يقوم Access بإنشاء عبارة SQL المرادفة. يمكنك ملاحظة عبارة SQL لاستعلام عن طريق اختيار View ⇦ SQL أو بتحديد SQL View من قائمة زر Query View في شريط الأدوات عندما تكون في أسلوب عرض Design.يوضح شكل١٣ــ١٧ استعلام أسلوب عرض Design وأسلوب عرض SQL لاستعلام qrySortCompany. طلب اللغة الإنجليزية للبيانات هو "اسرد OrderID والبلد واسم الشركة، وتاريخ الأمر لكل الأوامر التي تم تعيينها حاليا لموظف تم اختياره في مربع التحرير والسرد cboCurrent، في نموذج frm AssignOrder تم فرزه بواسطة اسم الشركة".





الشكل ١٣-١٧ أسلوب عرض Design وأسلوب عرض SQL لاستعلام.

ملاحظة تتطلب معظم إجراءات VBA مزيجاً من الاستعلامات المخزنة وعبارات SQL، وتعلم كتابة عبارات SQL مباشرة مهارة هامة ومفيدة. وبينما أنت تستمر في عملك مع قواعد البيانات العلاقية. ستطور مهارات SQL مع ذلك لا تحتاج للتغلب على SQL فوراً بدلاً من ذلك، يمكنك التمهل في استخدام SQLباستخدام استعلام أسلوب عرض Design لإنشاء الاستعلام "ألا إذا كنت تنشئ استعلام SQL بعينه". بعد أن تقوم بإنشاء المسودة، بدل لأسلوب عرض SQL، انسخ عبارة أو جملة SQL، بدل لإطار بدل لإطار Module والصق عبارة أو جملة SQL في إجراء VBA. اصعب نقطة هي تعديل عبارة أو جملة SQL لتتضمن متغيرات VBA ثم التعبير عن النتيجة كسلسلة نص يستطيع Jet فهمها.

فهم مفردات وقواعد اللغة الخاصة بلغة SQL

عبارة SQL هي طلب للبيانات من قاعدة بيانات علاقية تحتوي العبارة على أسماء الجداول والحقول التي تحمل البيانات ومعلومات إضافية تعرف البحث والإجراء الذي تريد حدوثه. SQL لها مفردات كلمات إنجليزية تتضمن حوالي ١٠٠ كلمة SQL أساسية. تظهر الكلمات الأساسية في SQL كأحرف كبيرة في أساليب عرض SQL رغم من أن كلمات SQL الأساسية ليست حساسة لحالة الأحرف. في Access تتضمن عبارة SQL فاصلات وجمل وعوامل تشغيل ومجموعة وظائف كلية. يوضح جدول ١٣-١ الأوامر السبعة في Access SQL التي تستطيع استخدامها لطلب إجراء معين. يوضح جدول ١٣-٢ معظم كلمات SQL الأساسية الأكثر شيوعاً

جدول ١٣-١ أوامر Access SQL

الرمز	الوصف
SELECT	يسترجع البيانات المخزنة من الجداول في قاعدة البيانات كمجموعة سجلات.
INSERT	يضيف سجلات جديدة للجدول.
DELETE	يحذف سجلات من جدول
UPDATE	يعدل حقولاً معينة في السجلات الموجودة.
CREATE	ينشئ جدولاً جديداً في قاعدة البيانات أو ينشئ فهرساً لحقل موجود أو مجموعة من الحقول.
ALTER	ينشئ حقولاً جديدة لجدول موجود أو يحذف حقولاً موجودة.
DROP	يحذف جدولاً موجوداً أو فهرساً موجوداً لحقل أو مجموعة من الحقول.

جدول ١٣-٢ الكلمات الأساسية الشائعة في SQL

الرمز	الوصف
DISTINCTROW	يحذف السجلات المكررة في النتيجة.
AS	تعيين الاسم المستعار الذي تريد استخدامه لاسم الحقل الذي يسبق الاسم المستعار.
FROM	يعين الجداول أو الاستعلامات التي تحتوي على الحقول التي تريدها في النتيجة.
WHERE	يعين المعيار الذي تريد استخدامه لتحديد السجلات. ويتبع WHERE ظرف بحث، تحتوي النتيجة على سجلات فقط يكون ظرف البحث لها True.
JOIN	يعين الربط بين السجلات في جداول أو استعلامات مختلفة، تستخدم JOIN مع الكلمات الأساسية INNER وLEFT وRIGHT لتحديد نوع الربط.

جدول ١٣-٢ الكلمات الأساسية الشائعة في SQL

الرمز	الوصف
ORDERBY	يعين ترتيب الفرز للنتيجة.
GROUPBY	يعين الحقل الذي تريد استخدامه لتكون مجموعات من الصفوف التي تم تحديدها. يرجع الاستعلام سجلاً واحداً لكل قيمة منفردة في الحقل الذي تم تعيينه.

تبدأ عبارة SQL بأمر من الأوامر وتتضمن جملة أو اكثر تعين البيانات التي توفر معلومات إضافية عن النتيجة النهائية التي تطلبها. تبدأ كل جملة بكلمة أساسية مثل WHERE أو ORDERBY أو FROM ويمكن أن تتضمن كلمات SQL أساسية أخرى أو وظائف مضمنة "لكن ليس وظائف معرفة من قبل المستخدم" أو جدولاً أو أسماء استعلام أو تعبيرات أو ثوابت ومراجع لتحكمات في النماذج أو التقارير. تستخدم Access SQL علامات الترقيم الموضحة في جدول ٢٣-٣.

جدول ١٣-٣ علامات ترقيم Access SQL

الرمز	الوصف
الفاصلة المنقوطة ";"	استخدام الفاصلة المنقوطة لإنهاء كل عبارة من عبارات SQL.
الفاصلة ","	استخدام الفاصلات لفصل الأسماء في القوائم مثل First Name, Last Name.
الأقواس المربعة "[...]"	استخدام الأقواس المربعة لتضمين اسم يحتوي على مسافات أو أحرف خاصة مثل [Cost / Unit] أو [Last Name].
فترة "."	استخدام الفترة لفصل اسم جدول أو الاستعلام عن اسم الحقل عندما يكون اسم الجدول أو الاستعلام مطلوبا لتعريف الحقل مثل Customers. Company Name.
أحرف Wildcard	استخدام Wildcards كأحرف نائبة لأحرف أخرى بمعامل تشغيل Like لإنشاء ظرف بحث Wildcard على سبيل المثال،? و* و# و[!].

جدول ١٣-٣ علامات ترقيم Access SQL

الرمز	الوصف
علامات التنصيص	استخدام علامات التنصيص المزدوجة """"والفردية "'" لتضمين قيم حرفية مثل "Germany" وللإشارة إلى وجود سلسلة داخل سلسلة.
رمز رقم "#"	استخدام رمز الرقم لتضمين قيم تاريخ ووقت حرفية مثل #1/2/99#.

بالرغم من أن كل استعلام تقوم بإنشائه في شبكة التصميم له عبارة SQL مقابلة، هناك بعض الاستعلامات تدعى استعلامات SQL أو استعلامات SQL معينة، تقوم بإنشائها فقط كعبارات SQL في أسلوب عرض SQL وهذه الاستعلامات ليس لديها مرادفات أسلوب عرض Design. تتضمن استعلامات SQL الأنواع الستة التالية:

Union queries: تستخدم لمزج نتائج تحديد استعلام أو اكثر في نتيجة واحدة عندما يكون لكل تحديد استعلام نف الأعمدة والحقول، المقابلة هلا نفس البيانات.

Non-equijoin queries: تستخدم إنشاء روابط لا تعتمد على المساواة.

Single-record append queries: تستخدم لتحديد قيمة كل حقل في سجل واحد جديد ثم تلحق السجل بجدول أو استعلام موجود.

Data definition queries: تستخدم لإنشاء أو تعديل أو حذف جدول أو إنشاء حذف فهرس في جدول.

Supqueries: تستخدم عندما يعتمد الاستعلام على نتيجة استعلام آخر.

Pass-through queries: تستخدم لإرسال أوامر تسترد السجلات أو تغير البيانات مباشرة لمزود قاعدة بيانات SQL.

استخدام استعلامات مخزنة ضد عبارات SQL "اعتبارات الأداء"

بعد إنشائك لاستعلام في أسلوب عرض Design أو SQL يمكنك القيام بواحد مما يلي:

- احفظه كاستعلام مخزن موجود في إطار Database وادخل اسم الاستعلام حيثما رغبت في استخدامه
- ادخل عبارة SQL في تعيين خاصية أو عبارة VBA

بالرغم من أنك تستطيع استخدام استعلام مخزن أو عبارة SQL في أي موقف يتطلب استعلاماً، في أغلب الأحوال، يعطي استخدام استعلام مخزن افضل أداء عندما تخزن استعلاماً ككائن قاعدة بيانات، يحلل Access الاستعلام ويحفظ إصداراً مثالياً. يحلله Access مرة أخرى في المرة التالية التي تشغله فيها ويخزن الإصدار الذي تم جعله مثالياً مؤخراً. من ناحية أخرى، في كل مرة تشغل عبارة SQL يحلل access العبارة ويحدد الطريقة المثلى لتنفيذها. ولأن التحليل والأمثلة يتطلبان وقتا، تنفذ عبارة SQL ببطيء اكثر من الاستعلام المخزن المرادف.

إنشاء استعلامات جديدة مخزنة في إجراءات VBA

الاستعلام هو مجموعة من الإرشادات أو التعريفات لاسترداد وتعديل البيانات. عندما تعمل تفاعلياً، فأنت تنشئ استعلاما جديداً وتحفظ ككائن إطار Database في قاعدة البيانات الحالية. عندما تعمل مع الاستعلامات مباشرة في VBA فأنت تشير إلى استعلام مخزن ككائن QueryDfes وهو واحد من كائنات تشغيل البيانات الذي يدار بواسطة Jet وكما يوضح الفصل السادس، كائن QueryDef وهو في مجموعة عضو في مجموعة QueryDefs "مجموعة لكل الاستعلامات المخزنة" وتنتمي مجموعة QueryDefs لكائن Database في قاعدة البيانات التي تم إنشاءها بواسطة محرك قاعدة بيانات Jet. "انظر الفصل السادس لمزيد من المعلومات حول العلاقة بين كائنات تشغيل البيانات".

ملاحظة من المميزات القيمة في Access 2000 هي أنك تستطيع تجنب محرك Jet والعمل مباشرة مع قاعدة بيانات ODBC باستخدام محرك قاعدة بيانات آخر مثل Microsoft SQL Server. في هذه الحالة، تنتمي مجموعة QueryDefs لكائن Connection بدلا ن Database. يفترض هذا الفصل أنك تستخدم محرك قاعدة البيانات Jet. انظر الفصل السادس لمزيد من المعلومات حول كائن Connection.

يمكنك إنشاء استعلامات مخزنة جديدة في إجراءات VBA باستخدام طريقة CreateQueryDef لكائن Database لأن هذه الطريقة تقوم بإنشاء كائن، يمكنك ان تعلن متغير كائن للإشارة إلى الكائن الجديد. بناء الجملة لإعلان المتغير وإنشاء كائن QueryDef جديد في قاعد البيانات هو:

```
Dim qdf As QueryDef, db As Database
Set qdf = db.CreateQueryDef"name,sqlstatement"
```

حيث:

- ♦ qdf متغير كائن يشير لكائن QueryDef الجديد الذي تنشئه.
- ♦ db متغير كائن يشير إلى كائن قاعدة بيانات مفتوح سيحتوي على QueryDefالجديد.
- ♦ Name تعبير سلسلة اختياري يتعرف بمسمى QueryDef الجديد.
- ♦ Sqlstatment عبارة SQL صالحة اختيارية يتم التعبير عنها كسلسلة.

إذا حددت اسماً صالحاً كوسيطة الأولى للطريقة، بحفظ Access تلقائياً كائن QueryDef الجديد كاستعلام إطار Database ويلحق الكائن بمجموعة QueryDefs.

إذا حذفت وسيطة في العبارة التي تنشئ الاستعلام الجديد، يمكنك استخدام خصائص Name وSQL لكائن QueryDef لتعريف الاستعلام ما يلي:

```
Set qdf = db.CreateQueryDef""
qdf.Name = name
qdf.SQL = sqlstatement
```

إذا حذفت وسيطة Name واستخدمت خاصية Name لتعريف الاستعلام، لايحفظ Access تلقائياً QueryDef الجديد، يجب أن تلحق كائن QueryDef لمجموعة QueryDefs كما يلي:

```
db.QueryDefs.Append qdf.Name
```

يمكنك إنشاء كائن QueryDef مؤقت بتعيين وسيطة name لسلسلة طولها صفر. ولأن السلسلة التي طولها صفر ليست اسماً صالحاً، لايمكن حفظ كائن QueryDef الجديد لقاعدة البيانات. لذلك عندما ينتهي الإجراء، يتوقف كائن QueryDef المؤقت عن الوجود.

كأمثلة، سنقوم بإنشاء بعض الاستعلامات الجديدة المخزنة. قم بإنشاء وحدة جديدة تدعى basQueryDef لهذه الأمثلة.

إنشاء تحديد استعلام جديد

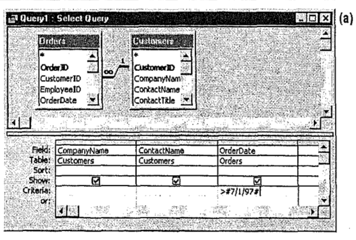
سنقوم بإنشاء استعلام جديد للبحث عن العملاء الذين وضعوا الأوامر بعد تاريخ معين.سنصمم استعلاماً في استعلام أسلوب عرض Design والصق عبارة SQL في إجراء VBA الجديد. افتح استعلام أسلوب عرض Design وقم بإنشاء استعلام يعتمد على جداول Orders وCustomers التي تقوم اسم الشركة واسم الاتصال وتاريخ الأمر لكل أمر تم وضعة بعد ٩٧/٢/٧. "يوضح شكل ٢٣-٢٨أ" أسلوب عرض Design للاستعلام.

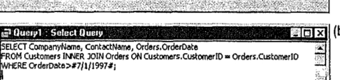
عندما تقوم بإنشاء استعلام في استعلام أسلوب عرض Design، يتضمن Access دائماً اسم الجدول لكل حقل ويمكن أن يحتوي على مجموعة وافرة من الأقواس في عبارة SQL المقابلة يقوم اسم الجدول مطلوباً فقط عندما يكون لديك حقول بنفس الاسم في جدول أو جدولين "أو الاستعلامات المضمنة في الاستعلام" في شكل "٢٣-٢٨ ب" الأقواس الوافرة وأسماء الجداول

الخيارية تم حذفها. عندما تزيل الأقواس الوافرة وأسماء الجدول، بـــدل لأســـلوب عـــرض SQL وشغل الاستعلام للتأكد من أن عبارة SQL أساسية ما زالت صالحة. انســـخ عبـــارة SQL إلـــى Clipboard. لا تغلق إطار Query لأننا سنستخدم هذا الاستعلام في المثال التالي.

ادخل إجراء NewstoredQuery. الموضح فيما يلي، في وحدة basQueryDefs سنســـمي الاستعلام المخزن الجديد QueryRecentCustomer عندما تلصق عبارة SQL الموضحة فـــي "شكل ١٣-١٨ب" ستحتاج لإعادة توصيل السطور الثلاثة في سطر شفرة واحـــد بعـــد إعـــادة توصيل القطع الخاصة بعبارة SQL، يمكنك إعادة فصلها لجعل الشفرة اكثر قابليـــة للقـــراءة. لا يتيح لك VBA تجميع القطع كما هو موضح في المثال الآتي. بعد إنشاء الاستعلام الجديد، يحـــدث الإجراء، إطار Database وينهي.

```
Public Sub NewStoredQuery""
   Dim db As Database, qdf As QueryDef, strSQL As String
   strSQL = "SELECT CompanyName, ContactName, OrderDate FROM "
   strSQL = strSQL & "Customers INNER JOIN Orders ON "
   strSQL = strSQL & "Customers.CustomerID = Orders.CustomerID "
   strSQL = strSQL & "WHERE OrderDate > #7/1/97#;"
   Set db = CurrentDB
   Set qdf = db.CreateQueryDef""qryRecentCustomers", strSQL"
   RefreshDatabaseWindow
End Sub
```





الشكل ١٣-١٨

يمكنك إنشاء معظـــم عبـــارات SQL بإنشاء الاســـتعلام رســـوميا "أ" وبالتبديل لأســـلوب عرض SQL "ب".



عند استخدام عبارات SQL في شفرة VBA، يمكنك فصل العبارة إلى قطـع لجعل الشفرة اكثر قابلية للقراءة.عين القطعة الأولى من عبارة SQL لمتغير سلسلة مثل strSQL ثم جمع القطعة الثانية لمتغير السلسلة وهكذا. يجـب أن تحتوي كل قطعة على مسافات في البداية أو النهاية بحيث التعبير المجمـع الأخير تماما عبارة SQL الأصلية مرة أخرى.

شغل الإجراء فـي إطـار Immediate ثـم بـدل إطـار Database يظـهر استعلام qryRecentCustomers في القائمة التي تم تحديثها حدد الاستـعلام وانقـره نقـرا مزدوجـا لتشغيله. ويوضح "شكل ١٣-١٩" ورقة البيانات للاستعلام الجديد الذي تم تخزينه. عندما تنتـهي، أحذف استعلام qryRecectCustomers.

الشكل ١٣-١٩

ورقـة البيانـات لاستعلام تم إنشـاؤه في الإجراء.

qryRecentCustomers : Select Query

Company Name	Contact Name	Order Date
Alfreds Futterkiste	Maria Anders	25-Aug-1997
Alfreds Futterkiste	Maria Anders	03-Oct-1997
Alfreds Futterkiste	Maria Anders	13-Oct-1997
Alfreds Futterkiste	Maria Anders	15-Jan-1998
Alfreds Futterkiste	Maria Anders	16-Mar-1998
Alfreds Futterkiste	Maria Anders	09-Apr-1998
Ana Trujillo Emparedados y helados	Ana Trujillo	08-Aug-1997
Ana Trujillo Emparedados y helados	Ana Trujillo	28-Nov-1997
Ana Trujillo Emparedados y helados	Ana Trujillo	04-Mar-1998
Antonio Moreno Taquería	Antonio Moreno	22-Sep-1997
Antonio Moreno Taquería	Antonio Moreno	25-Sep-1997
Antonio Moreno Taquería	Antonio Moreno	28-Jan-1998
Around the Horn	Thomas Hardy	16-Oct-1997

Record: 1 of 492

إنشاء استعلام إجراء جديد

للمثال التالي، سنحول تحديد الاستعلام الذي أنشأناه لاستعلام إجراء يقوم بإنشاء جدول جديد انقـر في الاستعلام وأختر Query>Malce Table Query ادخل tblRecentOrders كاسم الجـدول الجديد يوضح شكل ١٣ -٢٠" أسلوب عرض SQL لاستعلام صنع الجدول مع إزالة الأقـواس الوافرة وأسماء الجداول غير الضرورية تتضمن عبارة SQL لاستعلام صنع جدول جملة INTO tablename لتحديد اسم الجدول الجديد.

الشكل ١٣-٢٠

عبـارة SQL لاستعـلام صنـع جدول.

Query1 : Make Table Query

```
SELECT CompanyName, ContactName, OrderDate INTO tblRecentOrders
FROM Customers INNER JOIN Orders ON Customers.CustomerID = Orders.CustomerID
WHERE OrderDate>#7/1/1997#;
```

ادخل إجراء NewActionQuery الموضح فيما يلي في وحـــدة basQueryDefs. ينشــئ الإجراء كائن Querydef الجديد ثم يعين خاصية SQL لعبارة SQL.

```
Public Sub NewActionQuery""
   Dim db As Database, qdf As QueryDef, strSQL As String
   strSQL = "SELECT CompanyName, ContactName, OrderDate "
   strSQL = strSQL & "INTO tblRecentOrders "
   strSQL = strSQL & "FROM Customers INNER JOIN Orders ON "
   strSQL = strSQL & "Customers.CustomerID = Orders.CustomerID "
   strSQL = strSQL & "WHERE OrderDate > #5/1/98#;"
   Set db = CurrentDB
   Set qdf = db.CreateQueryDef""qryRecentCustomers""
   qdf.SQL = strSQL
   RefreshDatabaseWindow
```

شغل الإجراء في إطار Immediate. بدل لإطار Database ولاحـــظ اســتعلم الإجــراء الجديد.حدد استعلام الإجراء الجديد وانقر مرتين لتشغيله. يقوم Access بتشغيل الاستعلام الجديــد وينشئ الجدول. عندما تنتهي، احذف الاستعلام الجديد والجدول.

إنشاء استعلام مؤقت

كمثال أخير، سنقوم بإنشاء استعلام مؤقت يحدد سجلات لعملاء حاليين لا يكون الاستعلام الجديــد موجوداً عندما ينتهي الإجراء. ادخل إجراء NewTemporaryQuery الموضح فيما يلـــي فـــي basQuerydef يماثل هذا الإجراء، الإجراء في المثال السابق فيما عدا أن سلسلة طولها صفـــر مستخدمة كوسيطة اسم للاستعلام.

```
Public Sub NewTemporaryQuery""
   Dim db As Database, qdf As QueryDef, strSQL As String
   strSQL = "SELECT CompanyName, ContactName, OrderDate "
   strSQL = strSQL & "INTO tblRecentOrders "
   strSQL = strSQL & "FROM Customers INNER JOIN Orders ON "
   strSQL = strSQL & "Customers.CustomerID = Orders.CustomerID "
   strSQL = strSQL & "WHERE OrderDate > #5/1/98#;"
   Set db = CurrentDB
   Set qdf = db.CreateQueryDef""""""
   qdf.SQL = strSQL
   RefreshDatabaseWindow
End Sub
```

شغل الإجراء في إطار Immediate. عندما تشغل الإجراء، يقوم VBA بإنشاء الاستعلام الجديد لكن لا يستطيع حفظه لأن الاسم غير صالح. بدل لإطار Database ولاحظ ان قائمة الاستعلامات لم تتغير.

تشغيل تحديد استعلام مخزن

يسترد تحديد الاستعلام البيانات ويرجع مجموعة من السجلات للذاكرة. لتشغيل تحديد استعلامات مخزن، يمكنك استخدام طريقة OpenQuery لكائن Docmd أو طريقة OpenRecordSet.

استخدام طريقة OpenQuery لكائن Docmd

استخدم طريقة OpenQuery لكائن Docmd عندما تريد تشغيل تحديد استعلام أو استعلام جدولي وعرض إطار Query في واحد من أساليب عرضه. بناء الجملة لطريقة OpenQuery هو:

```
DoCmd.OpenQuery queryname,view,datamode
```

حيث:

- QueryName تعبير سلسلة واسم صالح للاستعلام في قاعدة البيانات الحالية.
- View ثابت اختياري حقيقي لتحديد أسلوب العرض "acNormal أو acDesign أو acPreview".
- DataMode ثابت اختياري حقيقي لتحديد نوع البيانات "acAdd أو acEdit أو acReadOnly" أسلوب العرض الفرضي هو أسلوب Normal ونوع البيانات الفرضي هو Edit

على سبيل المثال، لتشغيل qryRecentCustomers الذي يقوم إجراء NewStoredQuery بإنشائه، أضف هذه العبارة التي تنشئ الاستعلام:

```
DoCmd.OpenQuery "qryRecentOrders"
```

عندما تشغل الإجراء، ينشئ الإجراء الاستعلام المخزن الجديد ويشغله ويفتح ورقة البيانات لعرض السجلات.

استخدام طريقة OpenRecordSet

تستخدم طريقة OpenRecordSet لتشغيل تحديد استعلام مخزن وإعادة النتيجة كمجموعة سجل في الذاكرة بدلا من إطار Query. يمكنك استخدام طريقة OpenRecordSet لكائن Database كما يلي:

Set rst = db.OpenRecordset"queryname,type,options,lockedits"

حيث:

- Queryname هو اسم تحديد استعلام موجود.
- Type هو ثابت حقيقي اختياري يحدد نوع مجموعة السجل "dbOpenDynaset, dbOpenSnapshot, or dbOpenForwardOnly".
- Option هو مزيج اختياري من ثوابت الأعداد الصحيحة التي تستخدمها لتحديد خصائص مجموعة السجل مثل: acAdd, acEdit, or acReadOnly.
- Lockedits هو ثابت اختياري يحدد الإغلاق لمجموعة السجلات "لا تستطيع فتح مجموعة سجل نوع جدول في الاستعلام".

ملاحظة في بقية هذا الفصل، db هو متغير كائن من نوع Database يشير لقاعدة بيانات مفتوحة. Rst هو متغير كائن من نوع Recordset. Qdf هو متغير كائن من نوع QueryDef. أيضا، تمثل sqlStatement عبارة SQL صالحة معبر عنها كسلسلة.

تستطيع أيضا إنشاء متغيرات كائن منفصلة للاستعلام ولمجموعة السجل واستخدام طريقة OpenRecordset لكائن QueryDef كما يلي:

Set qdf = db.QueryDefs"queryname"

Set rst = qdf.OpenRecordset"type,options,lockedits"

كمثال يمكن تعديل إجراء NewStoredQuery لتشغيل استعلام qryRecentCustomers وإعادة النتائج للذاكرة دون عرض ورقة بيانات. لفعل ذلك، أعلن rst كمتغير كائن لكائن Recordset الذي تم إنشائه في الذاكرة وأضف عبارات لإنشاء مجموعة سجل كما يلي: Dim rst As Recordset

Set rst = db.OpenRecordset ""qryRecentCustomers""

عندما تشغل الإجراء، يتمك إنشاء الاستعلام المخزن ومجموعة السجل. يتم تدمير مجموعة السجل عندما ينتهي الإجراء

تشغيل عبارة SQL

عندما لا تحتاج لتخزين تحديد استعلام أو عرض إطار Query فأنت لا تحتاج لإنشاء كائن QueryDef. في الواقع يمكن فقط تشغيل عبارة SQL لإنشاء مجموعة السجل في الذاكرة

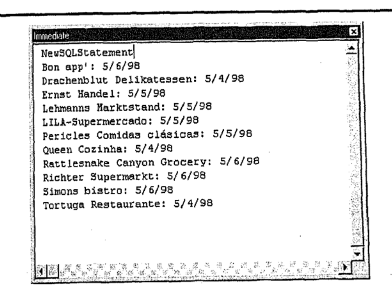
باستخدام طريقة OpenRecordset لكائن Database كما يلي:

```
Set rst = db.OpenRecordset"sqlstatement"
```

على سبيل المثال،يقوم إجراء New SQLStatment الموضح فيما يلي بإنشاء مجموعة سجل للعمل مع البيانات للملاء الحاليين دون إنشاء كائن Query. يطبع الإجراء تاريخ الترتيب لكل سجل تم تحديده لإطار Immediate:

```
Public Sub NewSQLStatement""
   Dim db As Database, rst As Recordset, strSQL As String
   strSQL = "SELECT CompanyName, ContactName, OrderDate "
   strSQL = strSQL & "FROM Customers INNER JOIN Orders ON "
   strSQL = strSQL & "Customers.CustomerID = Orders.CustomerID "
   strSQL = strSQL & "WHERE OrderDate > #5/1/98#;"
   Set db = CurrentDB
   Set rst = db.OpenRecordset"strSQL"
   Do Until rst.EOF
      Debug.Print rst!CompanyName & ": " & rst!OrderDate
      rst.MoveNext
   Loop
 End Sub
```

لاختبار الإجراء، أدرجه في وحدة QueryDef وشغله في إطار Immediate يتم طبع نتائج عبارة SQL انظر "شكل ١٣-٢١".



الشكل ١٣-٢١

عندما لا تحتاج لتخزين تحديد استعلام ولا تحتاج لعرض إطار استعلام، يمكنك تشغيل عبارة SQL لإنشاء مجموعة السجل في الذاكرة.

فرز وتصفية مجموعة سجل

عندما تريد فرز وتصفية مجموعة سجل، يمكنك استخدام تقنية SQL أو خصائص Filter وSort لمجموعه السجل أو خاصية Index لمجموعة سجل نوع جدول.

استخدام تقنيات SQL لفرز وتصفية مجموعة سجل

إذا كان بإمكانك تعريف مجموعة من السجلات باستخدام عبارة SQL، فإن أسرع طريقة لفرز وتصفية السجلات هي تعديل عبارة SQL لتتضمن جملا للفرز والتصفية واستخدام عبارة SQL المعدلة وتعديل أو فرز أو تصفية السجلات معاً.كمثال، يستخدم إجراء SQLSortFilter التالي عبارة SQL لتحديد السجلات من جدول Customers ولتصفية السجلات للحصول على عملاء المانيا فقط ثم فرز السجلات باسم الشركة:

```
Public Sub SQLSortFilter""
   Dim db As Database, rst As Recordset
   Set db = CurrentDB
   Set rst = db.OpenRecordset""SELECT * FROM Customers WHERE " _
      & "Country = 'Germany' ORDERBY CompanyName""
   Do Until rst.EOF
      Debug.Print rst!CompanyName
      rst.MoveNext
   Loop
End Sub
```

يطبع الإجراء CompanyName من مجموعة السجل التي تم فرزها وتفتيتها في إطار Immediate. يمكنك إدخال الإجراء في وحدة جديدة تدعى basGroups وتشغيل الإجراء في إطار Immediate.

استخدام خصائص Filter وSort لمجموعة سجل

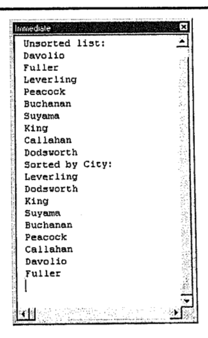
يمكنك استخدام خصائص Filter وsort لفرز وتصفية مجموعة سجل موجودة، عند استخدامك لهذه الخصائص، لا تتأثر مجموعة السجل الموجودة ويجب عليك إنشاء مجموعة سجل ثانية تعتمد على الأولى حتى ترى النتيجة المترتبة على التعيينات.

يمكنك استخدام خاصية Sort لفرز مجموعة سجلات مجموعة حيوية أو مجموعة سجل من نوع snapshot-type. تعيين خاصية Sort لتعبير سلسلة هو جملة OrderBy لعبارة SQL صالحة دون عبارة OrderBy. لا يمكنك استخدام خاصية Sort لفرز مجموعة سجل نوع جدول.

يقوم إجراء DAOSort التالي بإنشاء مجموعة سجل نوع مجموعة حيوية لجدول Customers. يعين خاصية Sort لفرز السجلات بالبلد وينشئ مجموعة سجل أخرى تعتمد على المجموعة الأولى التي تتضمن تأثير الفرز

```
Public Sub DAOSort""
   Dim rst As Recordset, rstSort As Recordset
   Set rst = CurrentDB.OpenRecordset""Employees", dbOpenDynaset"
   rst.Sort = "City"
   Set rstSort = rst.OpenRecordset""
   Debug.Print "Unsorted list:"
   Do Until rst.EOF
      Debug.Print rst!LastName
      rst.MoveNext
   Loop
   Debug.Print "Sorted by City:"
   Do Until rstSort.EOF
      Debug.Print rstSort!LastName
      rstSort.MoveNext
   Loop
End Sub
```

ويستخدم هذا الإجراء حلقات Do Until لطباعة أسماء الشركات لكل من مجموعات السجلات المخزنة وغير المخزنة "انظر الشكل ١٣-٢٢".



الشكل ١٣-٢٢

عندما تستخدم خاصية Sort لمجموعة سجل تحتاج لإنشاء مجموعة سجل أخرى تعتمد على الأخرى لرؤية تأثير الفرز.

استخدام خاصية Index لفرز مجموعة سجل نوع جدول

بتعيين خاصية Index الخاصة بها لفهرس موجود للجدول. يجب أن تستخدم خاصية Index لفرز مجموعة نوع جدول ولا تستطيع استخدام خاصية Sort. يقوم إجراء DaoSortTable بفرز مجموعة سجل نوع جدول:

```
Public Sub DAOSortTable""
   Dim rst As Recordset, rstSort As Recordset
   Set rst = CurrentDB.OpenRecordset""Customers", dbOpenTable"
   Debug.Print "Sorted by Primary Key:"
   Do Until rst.EOF
      Debug.Print rst!City & " " & rst!CompanyName
      rst.MoveNext
   Loop
   rst.Index = "City"
   rst.MoveFirst
   Debug.Print "Sorted by City:"
   Do Until rst.EOF
      Debug.Print rst!City & " " & rst!CompanyName
      rst.MoveNext
   Loop
End Sub
```

يبدأ هذا الإجراء بإنشاء مجموعة سجل نوع جدول لنوع جدول Customers وطبع اسم البلد والشركة في ترتيب المفتاح الأولى. يعد فهرس الإجراء الترتيب الحالي لفرز السجلات بالبلد "واحد من الفهارس الموجودة للجدول" يحرك مؤشر السجل الحالي للسجل الأول مرة أخرى ويطبع أسماء البلد والشركة لمجموع السجلات التي تم فرزها في إطار Immediate.

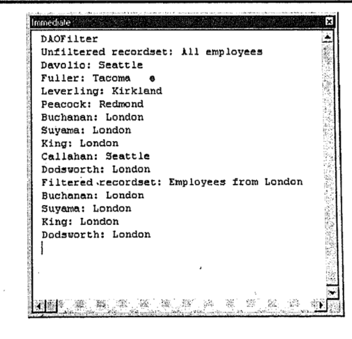
تصفية مجموعة سجل نوع مجموعة حيوية أو نوع لقطة سريعة

تستخدم خاصية Filter لتصفية السجلات في مجموعة سجل نوع المجموعة الحيوية أو نوع اللقطة السريعة ولا تستطيع تصفية السجلات الخاصة بمجموعة سجل نوع جدول. عين خاصية Filter لمجموعة السجل إلى تعبير سلسلة يكون جملة WHERE لعبارة SQL دون كلمة WHERE.بعد تعيين خاصية Filter، يجب أن تنشئ كائن Recordset ثاني يعتمد على الأول لرؤية نتيجة التصفية.

يقوم إجراء DAOFilter التالي بتصفية نتيجة سجل نوع مجموعة حيوية.

```
Public Sub DAOFilter()
    Dim rst As Recordset, rstFilter As Recordset
    Set rst = CurrentDB.OpenRecordset("Employees", dbOpenDynaset)
    rst.Filter = "City = 'London'"
    Set rstFilter = rst.OpenRecordset()
    Debug.Print "Unfiltered recordset: All employees"
    Do Until rst.EOF
        Debug.Print rst!LastName & ": " & rst!City
        rst.MoveNext
    Loop
    Debug.Print "Filtered recordset: Employees from London"
    Do Until rstFilter.EOF
        Debug.Print rstFilter!LastName & ": " & rstFilter!City
        rstFilter.MoveNext
    Loop
End Sub
```

يفتح هذا الإجراء مجموعة سجل نوع مجموعة حيوية في جدول Employee ويعين التصفية لتحديد الموظفين من لندن ثم ينشئ مجموعة سجل ثانية للسجلات التي تم تصفيتها. يطبع الإجراء الاسم الأخير للسجلات من مجموعة السجلات التي تمت تصفيتها والتي لم تتم تصفيتها "انظر الشكل "١٣-٢٣".



الشكل ١٣-٢٣

عندما تستخدم خاصية Filter لمجموعة سجل، يجب أن تنشئ مجموعة سجل جديدة تعتمد على المجموعة الأصلية لرؤية تأثير التصفية.

إعادة استخدام متغير الكائن

تنشئ الإجراءات DAOSort وDAOSort Table وDAOFilter متغير كائن ثـــاني لمجموعـــة السجل التي تمت تصفيتها أو فرزها بحيث يمكن استخدام كلاً من مجموعات السجل الأصليـــة أو تلك التي تمت تصفيتها وفرزها في الإجراء. على سبيل المثال، للطبع لإطـــار Immediate. إذا لم تحتاج للعمل مع مجموعات السجل كليهما، يمكنك تجنب عبء الذاكرة لمجموعتي سجل بإعادة استخدام متغير الكائن.

يستخدم البرنامج التالي متغير كائن rst ويوضح كيفية دمج الفرز والتصفية:

```
Set rst = db.OpenRecordset"sqlstatement"
rst.Sort = strSortCondition
rst.Filter = strFilterCondition
Set rst = rst.OpenRecordset""
```

تشغيل استعلامات إجراء مخزنة

يعدل استعلام الإجراء البيانات في قاعدة البينات لكنه لا يرجع لأي سجلات في الذاكرة. لتشـــغيل استعلام إجراء مخزن في إجراء VBA، يمكنك استخدام طريقة OpenQuery لكائن Docmd أو طريقة Execute.

استخدام طريقة OpenQuery لكائن Docmd

بناء الجملة لطريقة OpenQuery لتشغيل استعلام إجراء هي:

```
DoCmd.OpenQuery queryname
```

Queryname: هو تعبير سلسلة لأسم الاستعلام المخزن في قاعدة البيانات الحالية.

على سبيل المثال، يمكنك تشغيل استعلام صنع الجدول qryRecentCustomers الذي تـــم إنشائه في مثال NewActIonQuery عن طريق إضافة العبارة التالية بعد العبارات التي تعـــرف الاستعلام:

```
DoCmd.OpenQuery "qryRecentCustomers"
```

عندما تشغل الإجراء المعدل. يتم إنشاء كلاً من استعلام صنع الجدول والجدول.

تلميح إذا كنت لا تريد عرض رسائل التأكيد الفرضية التي لا يعرضـــها Access عندما تشغل استعلام إجراء، استخدم طريقـــة SetWarnings فـــي False لإغلاق الرسائل قبل أن تشغل استعلام الإجراء عندما تغلق تحذيرات النظام في إجراء ما، يجب أن تعيدها بتعيين وسيطة WarningSon في True.

استخدام طريقة Execute

لكل من كائن Query Def وDatabase طرق Execute تستطيع استخدامها لتشغيل استعلام إجراء تم تخزينه. بناء الجملة لطريقة Execute لكائن Query Def هو:

qdf.Execute options

لكائن Database، بناء الجملة هو: db.Execute source, options

Source: هو تعبير سلسلة وهو اسم الاستعلام الذي تم تخزينه "أو عبارة "SQL. في العبارتين كلتيهما، وسيطة Options هي ثابت عدد صحيح اختياري يحدد خصائص الاستعلام ويتضمن ما يلي:

DbDenyWrite	يتجاهل اسم الكتابة للمستخدمين الآخرين.
DbInconsistent	ينفذ تحديدات غير مترابطة
DbConsistent	ينفذ تحديدات مترابطة
DbSQLPassThrough	يؤدي بعبارة SQL إلى التمرير لقاعدة بيانات ODBCللتشغيل
DbFailOnError	يعيد التحديثات إذا حدث خطأ ما
DbSeeChanges	يولد خطأ إذا غير مستخدم آخر البيانات التي تحررها
DbRunAsync	ينفذ الاستعلام لا تزامنياً فقط مع ODBCDirect
DbExecDirect	ينفذ العبارة دون استدعاء وظيفة SQL Prepare أولاً "ODBC" فقط"



إذا كانت أية سجلات مغلقة عندما تشغل طريقة Execute لاستعلام Update أو Delete، لن تحدث الطريقة أو تحذف السجلات المغلقة. مع ذلك لا تفشل طريقة Execute ولا يوجد دليل على السجلات المغلقة. ولتجنب عدم ترابط البيانات الذي تتسبب فيه السجلات المغلقة، استخدم دائما خيار dbFileOnErorre لاستعلام Update أو Delete لسحب كل التغييرات الناجحة إذا كانت أي من السجلات المتأثرة باستعلام الإجراء مغلقة.

كمثال، ما يلي إصدار معدل لإجراء NewActrionQuery الذي يستخدم طريقة Execute لتشغيل استعلام صنع جدول.

```
Public Sub NewActionQuery""
   Dim db As Database, qdf As QueryDef, strSQL As String
   strSQL = "SELECT CompanyName, ContactName, OrderDate "
   strSQL = strSQL & "INTO tblRecentOrders "
   strSQL = strSQL & "FROM Customers INNER JOIN Orders ON "
   strSQL = strSQL & "Customers.CustomerID = Orders.CustomerID "
   strSQL = strSQL & "WHERE OrderDate > #5/1/98#;"
   Set db = CurrentDB
   Set qdf = db.CreateQueryDef""qryRecentCustomers""
   qdf.SQL = strSQL
   qdf.Execute dbFailOnError
   RefreshDatabaseWindow
End Sub
```

لاختبار هذا الإجراء، عدل إجراء NewActionQuery كما شرح فيما سبق، ثم بدل لإطار Database واحذف qryRecentCustomers وtblRecentOrders إذا لزم الأمر شغل الإجراء في إطار Immediate.

ملاحظة استخدم خاصية RecordeAffected لكائن QueryDef أو Database لتحديد عدد السجلات التي تأثرت بطريقة Execute.

تشغيل عبارة SQL لاستعلام إجراء أو استعلام تعريف بيانات

يمكنك إنشاء عبارات SQL لاستعلامات الإجراء ولاستعلامات تعريف بيانات. يمكنك إنشاء عبارات SQL لأنواع استعلامات الإجراء الأربعة التي تعدل الجدول بتصميم الاستعلام في استعلام أسلوب عر Design وبالتبديل لأسلوب عرض SQL. يوضح "جدول ١٣-٤" أوامر SQL وأمثلة لاستعلامات الإجراء الأربعة. يتضمن الجدول أيضاً مثالاً لاستعلام SQLSpecific لإلحاق سجل.تدعى أوامر SQL لاستعلام الإجراء أوامر لغة استغلال البيانات "DML".

الجدول ١٣-٤: أوامر SQL لاستعلام الإجراء

نوع الاستعلام	أمر SQL	مثال	الوصف
إلحاق	INSERT INTO SELECT	IINSERT INTO Customers FROM tblNewCustomers;	ينسخ سجلاً موجوداً من جدول أو استعلام آخر ويضيفها لجدول أو استعلام في قاعدة البيانات نفسها أو في قاعدة بيانات أخرى.
إلحاق سجل واحد	INSERT INTO VALUSE	IINSERT INTO Categories (Category Name) VALUES ("Candies");	يضيف سجلاً بقيم معينة لجدول أو استعلام موجود.
احذف	DELETE	DELETE * FROM Customers WHARE Order Date < #1/1/90#;	يزيل السجلات من جدول أو اكثر.
تحديث	UPDATE...SET	UBDATE Orders SET OrderAmount = OrderAmount*1.02 WHERE ShipCountry = 'Germany';	يغير القيم في حقول معينة في جدول.
اصنع جدولاً	SELECT...INTO	SELECT * INTO tblRecentOrders FROM Orders WHERE OrderDate > #1/1/99#;	ينشئ جدولاً جديداً وينسخ السجلات الموجودة من جدول آخر في نف قاعدة البيانات.

يمكنك أيضاً استخدام عبارات SQL لإنشاء استعلام SQL بالذات SQL-Specific لتعريف وتعديل الجداول الجديدة والحقول والفهارس. تدعى هذه الاستعلامات استعلامات تعريف البيانات

وتستخدم أوامر تدعى أوامر لغة تعريف البيانات "DDL" يوضح "جدول ١٣-٥" الأوامر والأمثلة. لا تنتج استعلامات تعريف البيانات مجموعات سجل، لذا فأنت تشغلها بنفس الطريقة التي تشغل بها عبارات SQL لاستعلامات الإجراء.

الجدول ١٣-٥: أوامر SQL لاستعلامات تعريف البيانات

أمر SQL	مثال	الوصف
Create Table. أنشئ جدولاً	Creates new tables and fields.	ينشئ حقولاً وجداولاً جديدة
Create Index أنشئ فهرساً	CREATE INDEX	ينشئ فهرساً جديداً على جدول موجود بالفعل
Drop Table اسقط جدولاً	DROP TABLE tblRecentOrders;	يحذف جدولاً جديداً
Drop Index اسقط فهرساً	DROP INDEX Country ON Customers;	يحذف فهرساً موجوداً على جدول
Alter Table غير الجدول	Allter TABLE EMPLOYEE ADD COLUMN Salary currency;	يضيف أو يعدل أو يزيل عمود أو فهرس في جدول موجود.

ملاحظة عندما تريد حذف كل السجلات من جدول يمكنك استخدام استعلام Delete أو عبارة DROP Table عندما تستخدم استعلام Delete تحذف البينات فقط ويبقى تعريف الجدول، لكن، عندما تستخدم عبارة DROP Table يزيل تعريف الجدول من قاعدة البينات.

AccessSQL له أنواع البينات الخاصة به الموجودة في جدول "١٣-٦".

الجدول ١٣-٦: أنواع بيانات Access SQL

نوع البيانات	حجم التخزين	الوصف
BINARY	١ بايت	أي نوع بيانات.
BOOLEAN	١ بايت	قيم Yes / No.
BYTE	١ بايت	عدد صحيح بين ٠ و٢٢٥.
COUNTER	٤ بايت	رقم تتم زيادته عن طريق Jet عند إضافة سجل جديد "نوع البيانات Long".
CURRENCY	٨ بايت	نقطة ثابتة تستخدم حتى أربعة أماكن عشرية يمين النقطة العشرية.
DATETIME	٨ بايت	قيم تاريخ أو وقت
GUID	١٢٨ بايت	رقم ID فريد يستخدم في استدعاء الإجراء البعيد.
SINGLE	٤ بايت	دقة فريدة قيمة فاصلة عائمة.
DOUBLE	٨ بايت	عدد صحيح قصير بين ٣٢،٧٦٨ و٣٢،٧٦٨.
SHORT	٢ بايت	عدد صحيح طويل.
LONG	٤ بايت لكل حرف	٠ : ١٠٢ جيجابايت.
LONGTEXT	١ بايت	يستخدم لكائناتOLE.
LONGBINARY	١ بايت لكل حرف.	أحرف ٥ : ٢٢٥.
TEXT	يختلف	مرادف لنوع بيانات Varian.

استخدام طريقة RunSQL لكائن DoCmd

يمكنك تشغيل عبارة SQL لاستعلام إجراء أو استعلام تعريف بيانات باستخدام طريقة RunSQL لكائن DoCmd بناء الجملة هو:

```
DoCmd.RunSQL sqlstatement
```

أقصى طول لسلسلة SqlStatement هو ٣٢،٧٦٨ حرفاً.على سبيل المثال، يغير إجراء DDLAlterTable جدولاً بإضافة حقل Salary بنوع بيانات CURRENCY لجدول Employees.

```
Public Sub DDLAlterTable""
    Dim strSQL As String
    strSQL = "ALTER TABLE Employees ADD COLUMN Salary
CURRENCY;"
    DoCmd.RunSQL strSQL
End Sub
```

لاختبار الإجراء ادخله في وحدة basGroups وشغله في إطار Immediate افتح جدول Employees ولاحظ حقل Salary الجديد.

استخدام طريقة Execute لكائن Database

يمكنك استخدام طريقة Execute لكائن Database لتشغيل عبارة SQL لإجراء أو استعلام تعريف بيانات كما يلي:db.Execute sqlstatement, options

Options: مزيج اختياري للثوابت التي تحدد سلامة البيانات الخاصة بالاستعلام. على سبيل المثال، يمكنك حذف السجلات في جداول TableRecentCusyomers باستخدام إجراء DeleteRecords.

```
Public Sub DeleteRecords""
    Dim db As Database
    Set db = CurrentDB
    db.Execute "DELETE * FROM tblRecentCustomers;"
End Sub
```

يمكنك إزالة جدول من قاعدة البيانات باستخدام إجراء DDLDropTable.

```
Public Sub DDLDropTable""
    Dim db As Database
    Set db = CurrentDB
    db.Execute "DROP TABLE tblRecentCustomers;"
End Sub
```

إعادة تعيين مجموعات من السجلات

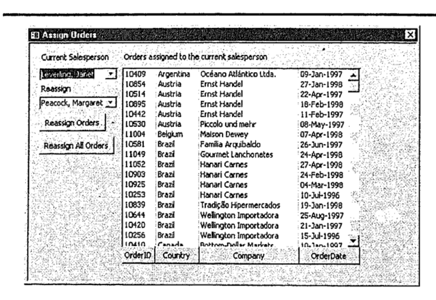
سنستخدم التقنيات الموضحة في الأقسام السابقة لتعديل نموذج frmAssignOrders المنشأة في قسم "استخدام مربع قائمة متعدد التحديد لتصفية السجلات". هذه المرة، سنضيف القدرة على إعادة

تعيين كل أوامر الموظف لموظف آخر. يستخدم المثال إجراء حدث cmdAsignAll التالي:

```
Private Sub cmdAssignAll_Click""
    Dim db As Database, rst As Recordset
    Dim varNumber As Variant, strSQL As String, str As String
    If IsNull"cboCurrent" Or IsNull"cboNew" Then
        str = "You must select a current employee and another "
        str = str & "employee you want to reassign orders to."
        MsgBox str
        Exit Sub
    End If
    Set db = CurrentDB
    strSQL = "UPDATE Orders SET EmployeeID = " & cboNew
    strSQL = strSQL & " WHERE EmployeeID = " & cboCurrent
    db.Execute strSQL, dbFailOnError
    cboCurrent = cboNew
    cboNew = Null
    lstOrders.Requery
End Sub
```

يبدأ هذا الإجراء باختبار مربعات التحرير والسرد الخاصة بالموظف. ويعرض رسالة وينهي إذا كانت مربعات التحرير خالية إذا تحديد الموظفين كلاهما، يشغل الإجراء عبارة SQL لتغيير حقل EmployeeID للأوامر المعروضة في مربع القائمة من الموظف الحالي إلى الموظف الجديد بعد تشغيل عبارة SQLK يعرض الإجراء مجموعة الأوامر الخاصة بالموظف الجديد وتعيين مربع التحرير والسرد الخاص بالموظف الجديد في Null، وإعادة استعلام مربع القائمة.

افتحح نموذج frmAssignOrders في أسلوب عرض Design وأضف زر أمر يدعى cmdAssignAll الموضح سابقاً لخاصية حدث زر OnClick. احفظ النموذج وبدل لأسلوب عرض Form اختر Janet leverling كمندوب المبيعات الجديد "انظر شكل ١٣-٢٤" انقر زر AssignAllOrders يتم إعادة تعيين كل أوامر Janet Leverling إلى Margaret Peacock.



الشكل ١٣-٢٤
ينفذ زر Assign AllOrders عبارة SQL لإعادة تنفيذ أوامر Ms.Leverling إلى Ms. Peacock

استخدام المعاملات

المعاملات هي مجموعة من التغيرات التي تريد معامليها كعملية واحدة عندما تعرف معاملاً ما، يتعقب متغير Jet التغيرات التي حدثت ويخزن التغييرات في قاعدة بيانات مؤقتة في الذاكرة حتى تتم كل تغييرات المعاملات."إذا لم تكن هناك ذاكرة كافية،يقوم Jet بإنشاء قاعدة البينات المؤقتة في القرص الثابت الخاص بالحاسب الآلي". إذا حدث خطأ ما أثناء تشغيل مجموعة التغييرات لا يتم تحديث أي من التغييرات لقاعدة البيانات إذا لم يحث خطأ، عند تمام حدوث أخر تغيير، يتم تحديث مجموعة التغييرات بأكملها لقاعدة البيانات الحالية.

المعاملات طريقة مهمة للحصول على سلامة البيانات عندما تقوم بمجموعة من التغييرات للبيانات الخاصة بك. على سبيل المثال، في عملية أرشيف، يتم إلحاق كل السجلات لجداول الأرشيف ثم حذفها من الجداول الحالية. إذا حدث فشل للقوة بين الاستعلامين تتواجد السجلات القديمة فبي جداول الأرشيف والجداول الحالية بتشغيل الاستعلامات في معاملات يمكنك تجنب هذا النوع من عدم ترابط البينات.

يستخدم Jet نوعين من المعاملات الضمنية والواضحة. عندما تشغل استعلام واحد سواء كاستعلام مخزن أو كعبارة SQL يشغل Jet الاستعلام ويقوم بكل التغييرات الضرورية في قاعدة بيانات مؤقتة في الذاكرة كمعاملات ضمنية. عند حدوث كل التغييرات. يقوم Jet بتحديث الجداول المتعلقة بالأمر. عندما تفشل القوة في أثناء تشغيل الاستعلام، عندما تفتح قاعدة البيانات بعد ذلك يمكن أن يبلغك Access بأن قاعدة البيانات تحتاج للإصلاح. عندما تصلح قاعدة البيانات يلغي Jet أية تغييرات جزئية حدثت أثناء فشل القوة ، على العكس عندما تريد تشغيل اكثر من استعلام أو عبارات عديدة كمعاملات يجب أن تستخدم عبارات عديدة كمعاملات يجب أن تستخدم عبارات تعرف معاملة واضحة.



تتيح لك خاصية Use Transaction التي تم تقديمها في Access 97،تحديد ما إذا كان Jet يشغل استعلام إجراء واحد كمعاملة ضمنية فرضيا، تعيين خاصية Use Transaction هو Yes أو True ويتم تشغيل استعلام الإجراء كمعاملة واضحة إذا كان استعلام الإجراء هو إجراء Delete أو إجراء Update يجب أن يغلق Jet عدد كبير من السجلات حتى تحدث كل التغييرات. تعيين خاصية Use Transaction في No في ورقة خاصية الاستعلام أو في False في إجراء VBA يمكن أن يتسبب في تحسين أداء حقيقي لأن Jet لا يحتاج لتخزين النتائج الوسيطة في قاعدة بيانات مؤقتة و لا يحتاج لإغلاق عدد كبير من السجلات. Transaction هي خاصية معرفة التطبيق" انظر الفصل ١٤ لمعلومات عن مثل هذه الخصائص ويطبق فقط على استعلام الإجراء.

يصف الفصل السادس كائن Jet WorkSpace ككائن تشغيل بيانات يدير المعاملات. يمكنك استخدام طرق كائن Jet WorkSpace لبدء وإنهاء معاملة ما يلي:

- تبدأ طريقة Begin Trans معاملة جديدة.
- تنهي طريقة Commit Trans المعاملة الحالية ويحفظ التغييرات لقاعدة البيانات.
- تنهي طريقة Roll Back المعاملة الحالية ويسترجع قاعدة البيانات للحالة التي كانت فيها عندما بدأت المعاملة.

لاكتشاف المعاملات سنعدل إجراء من الإجراءات لتعيين أوامر من موظف لآخر. من الاستخدامات المهمة للمعاملات هي إعطاء المستخدم الخيار لإلغاء إلغاء السجلات.

عدل إجراء cmd AssignAllClick الذي يعين كل أوامر الموظف لموظف آخر كما يلي:

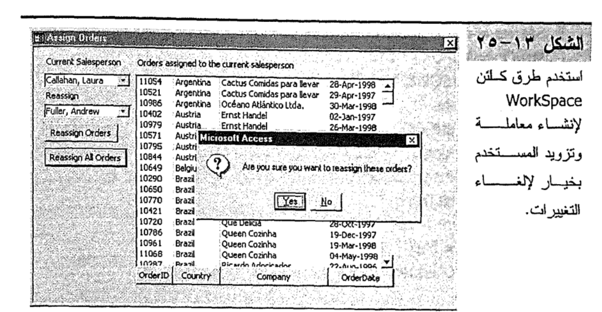
```
Private Sub cmdAssignAll_Click""
    Dim db As Database, rst As Recordset, strSQL As String
    Dim ws As Workspace, msg As String
    If IsNull"cboCurrent" Or IsNull"cboNew" Then
        msg = "You must select a current employee and another "
        msg = msg & "employee you want to reassign orders to."
        MsgBox msg
        Exit Sub
    End If
    Set db = CurrentDB
```

```
    Set ws = DBEngine.Workspaces"0"
    strSQL = "UPDATE Orders SET EmployeeID = " & cboNew
    strSQL = strSQL & " WHERE EmployeeID = " & cboCurrent
    ws.BeginTrans
    db.Execute strSQL, dbFailOnError
    msg = "Are you sure you want to reassign these orders?"
    If MsgBox "msg, vbQuestion + vbYesNo" = vbYes Then
       ws.CommitTrans
       cboCurrent = cboNew
       cboNew = Null
       lstOrders.Requery
    Else
       ws.Rollback
    End If
  End Sub
```

ينشئ الإجراء المعدل ws كمتغير كائن يشير لمساحة العمل الفرضية. تبدأ المعاملة قبل العبارة التي تنفذ عبارة SQL تماما لتحديث السجلات. بسبب عبارة ws.Bigen Trans. يتم الاحتفاظ بالتغييرات المطلوبة من قبل عبارة SQL في الذاكرة التي لم تكتب لقاعدة البيانات يسألك مربع الرسالة ما إذا كنت تريد حفظ التغييرات. إذا نقرت زر Yes, تحفظ طريقة Commit Trans التغييرات لقاعدة البيانات وإلا تتجاهل طريقة RollBack التغييرات.

احفظ النموذج وبدل لأسلوب عرض Form حدد مندوب مبيعات حالي وجديد وانقر زر أمر AssignAllOrders "انظر شكل ١٣-٢٥". انقر No لإعادة التغييرات.

ملاحظة لا تسمح كل قواعد البيانات بتعريف المعاملات. على سبيل المثال، لا تستطيع استخدام المعاملات على مجموعة سجل تعتمد على جدول Paradox. إذا لم يسمح كائن قاعدة بيانات أو مجموعة سجل بمعاملات، تفشل طرق Begin Trans وcommit Trans وrollback في هدوء دون توليد خطأ وقت تشغيل. وبسبب الفشل الذي يتم في هدوء، يجب أن تختبر قيمة خاصية Transaction لكائن Database أو كائن Recordset قبل استخدام طريقة Begin Trans إذا سمح كائن Database أو كائن Recordset بالمعاملات تكون قيمة خاصية Transaction في True وإلا كانت False.



الشكل ١٣-٢٥

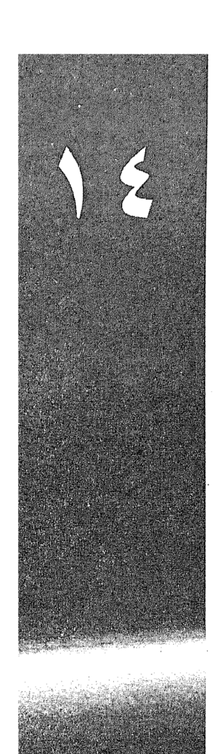
استخدم طرق كائن WorkSpace لإنشاء معاملة وتزويد المستخدم بخيار لإلغاء التغيرات.

خلاصة

قدم هذا الفصل الكثير من تقنيات VBA للعمل مع مجموعات من السجلات باستخدام طريقة النماذج وطريقة مجموعة السجل. وضح النصف الأول من هذا الفصل تقنيات لفرز وتحديد السجلات التي ستعرض في النموذج أو التقرير أما النصف الآخر، فناقش استخدام الاستعلامات المخزنة وعبارات SQL لتحديد مجموعات سجلات في الذاكرة دون الحاجة لعرض النتائج في ورقة بيانات أو نموذج أو تقرير ومن أهم ما جاء في هذا الفصل:

- يمكنك استخدام خصائص OrderBy وorderByOn لفرز سجلات نموذج أو تقرير. لكن يسهل استخدام متغير لحمل وإنشاء سلسلة فرز إجراء الفرز مع إجراءات VBA.
- يمكنك فتح نموذج أو تقرير بسجلات تم تحديدها باستخدام استعلام معلمة كمصدر السجل وباستخدام إجراء لتعيين خاصية RecordSource أو خاصية Filter وبتشغيل طرق OpenFrom وopenReport وapllyFilter.
- بعد فتح نموذج يمكنك تغيير تحديد السجلات المعروضة باستخدام التقنيات المضمنة FilterFor وfilterBySelection وFilterExcludingSelection وfilterByForm وباستخدام إجراء لتغيير خصائص النموذج وهي RecordSource وFilter وبتشغيل طرق OpenForm وopenReport وApllyFilter.
- يمكنك استخدام تقنية QueryByFrom لإنشاء واجهة تطبيق بحث تخصيص بسيطة.
- يتيح لك مربع قائمة متعدد التحديد بتحديد السجلات التي تفي بشرط البحث عشوائياً.

- يمكنك استخدام تقنيات التجول وتقنيات SQL للعمل مع مجموعة السجلات في مجموعة سجل.
- تستخدم تقنيات التجول طرقاً خاصة بكائن Recordset للتنقل خلال المجموعة والعمل على سجل واحد في المرة الواحدة.
- تستخدم تقنيات SQL استعلامات مخزنة وعبارات SQL لتعريف تحديد ما أو تعريف التعديلات التي تريد قاعدة البيانات Jet القيام بها تعتمد تقنيات SQL على شفرة مضمنة لمحرك Jet وتكون أسرع من تقنيات التنقل المقابلة.
- تستطيع إنشاء استعلامات جديدة "مخزنة أو مؤقتة" في إجراء VBA باستخدام طريقة CreatQueryDef.
- تستطيع إنشاء استعلامات تحديد باستخدام طريقة OpenQueryy عندما تريد عرض النتائج في ورقة بيانات أو باستخدام طريقة OpenRecordset عندما تريد فقط العمل مع النتائج في الذاكرة يمكنك أيضا استخدام طريقة OpenRecordset لتشغيل عبارة SQL عندما لا تحتاج لتخزين الاستعلام أو عرض النتائج.
- يمكنك فرز وتصفية مجموعة سجل باستخدام عبارة SQL لتعريف مجموعة السجل التي تم فرزها أو تصفيتها أو باستخدام خصائص Sort وFilter الخاصة بكائن Recordset.تحتاج لإنشاء مجموعة سجل جديدة تعتمد على مجموعة السجل الأصلية لرؤية تأثير الفرز أو التصفية.
- يمكنك تشغيل استعلامات إجراء مخزنة باستخدام طرق OpenQuery وexecute وتشغيل عبارات SQL لاستعلامات الإجراء باستخدام طرق RunSQL وExecute.
- تتيح لك طرق المعاملات الموجودة فقط في VBA تشغيل مجموعة من الاستعلامات أو العبارات كمجموعة.



إنشاء وتعديل كائنات قاعدة البيانات

يمكنك إنشاء جداول واستعلامات ونماذج وصفحات وصول إلى البيانات وتقارير بتفاعل في طرق عرض Design كجزء من عملية تصميم التطبيق. في أثناء إنشاء التطبيق، يتم إنشاء إجراءات VBA لاستخدام هذه الكائنات عند تشغيل المستخدم للتطبيق. ومع ذلك، يزودك Access VBA بتقنيات لإنشاء وتعديل معظم كائنات إطار Database برمجياً.

يوضح هذا الفصل كيفية إنشاء وتعديل الجداول والاستعلامات والنماذج وصفحات الوصول إلى البيانات والتقارير والوحدات النمطية باستخدام إجراءات VBA. سوف تتعلم أيضاً كيفية إنشاء خصائص وطرق مخصصة لهذه الكائنات.

تتكون الكائنات التي سنعمل معها من مكونين لأكسس هما: Access Application متضمناً النماذج والتقارير وصفحات الوصول إلى البيانات والوحدات النمطية بالإضافة إلى كائنات الوصول إلى بيانات محرك قاعدة البيانات متضمنة الجداول والاستعلامات. يحتوي نوع الكائن لكل مكون على تقنية فريدة لإنشاء وتعديل كائناته ولإنشاء طرق وخصائص مخصصة.

يقدم VBA أكسس العديد من طرق توجيه الكائن نحو البرمجة. ينتهي هذا الفصل بمناقشة أحد مزايا توجيه الكائن: إمكانية تعريف مخطط عمل نموذج أو تقرير وتكوين كائنات متعددة النماذج أو كائنات تقرير مخطط العمل.

فهم كيفية قيام أكسس ومحرك قاعدة البيانات بإنشاء كائنات

يعتبر كل من Access Application ومحرك قاعدة البيانات هما تطبيقات برامج منفصلة يعملان سوياً في علاقة أساسية لا يهتم المستخدم الفعال بها حيث يقوم مباشرة بالعمل مع تطبيقات أكسس.

تتضمن تطبيقات أكسس للبرمجة الداخلية كل اتصالات البرمجة لمحرك قاعدة البيانات. بالعمل مع واجهة تطبيق أكسس بفتح وإغلاق الجداول والاستعلامات والنماذج والتقارير والماكرو والوحدات النمطية سوف يعمل المحرك أو MSDE بشكل غير واضح على تكوين أو إتلاف الكائنات في الذاكرة وحفظ بعض الكائنات إلى ملف قاعدة البيانات.

تحتوي الكائنات الموجودة في نموذج كائن Access Application أو نموذج كائن DAO أو نموذج ADO على خصائص وطرق سبق تعريفها مما يجعل بإمكانك استخدام الكائنات برمجياً. لا يمكن استخدام الكائنات غير المضمنة في نموذج الكائن مثل الماكرو برمجياً. وبتشغيل كل من تطبيق أكسس ومحرك قاعدة البيانات معاً في نموذج الكائن الخاص به، لن تتمكن من معرفة التطبيق الذي ينتج النتائج التي تشاهدها على الشاشة. يمكننا مراجعة ما يحدث أثناء العمل مع واجهة تطبيق أكسس وملاحظة كائنات البرمجة التي يتم إنشاؤها وإتلافها آلياً. "قد تحتاج إلى الرجوع إلى نماذج الكائنات كما هو موضح في الفصل ٦".

وعند بدء تشغيل أكسس، في نموذج DAO، يتم تشغيل المحرك آلياً ويتم إنشاء كائن DBEngine جديد وكائن Workspace افتراضي باعتبارهما كائنات مؤقتة يتم إتلافها عند إغلاق إطار أكسس. وعند فتح قاعدة بيانات موجودة، يقوم المحرك بإنشاء كائن Database لتمثيل قاعدة البيانات الحالية باستخدام دالة المحرك DBEngine(0)(0) أو دالة أكسس CurrentDB يتعقب المحرك كائنات إطار Database التي تم إنشاؤها لقاعدة البيانات باستخدام كائنات Document. وعند أول فتح لقاعدة بيانات، يقوم المحرك بإنشاء كائنات Document لكل كائن تم حفظه في قاعدة البيانات الحالية وكائنات Container لكل فئة من الكائن المحفوظ في قاعدة البيانات الحالية، بما في ذلك ما يلي:

اسم "CONTAINER"	تحتوي على معلومات بشأن
الجداول Tables	الجداول والاستعلامات المحفوظة
العلاقات Relations	العلاقات المحفوظة
النماذج Forms	النماذج المحفوظة
التقارير Reports	التقارير المحفوظة
السيناريو Scripts	وحدات الماكرو المحفوظة
الوحدات النمطية Modules	الوحدات النمطية المحفوظة
صفحات الوصول إلى البيانات DataAccessPages	صفحات الوصول إلى البيانات المحفوظة

يحتوي كل كائن Container على معلومات إدارية عن فئته، على سبيل المثال من أنشأ الفئة ومن لديه تصريح لاستخدامها. يحتوي كائن Container للفئة على كائنات Document التي يتوفر لديها نفس المعلومات الإدارية عن كل كائن محفوظ في الفئة يقوم المحرك بإنشاء كائنات Database وcontainer وDocument ككائنات مؤقتة تمثل قاعدة البيانات وكائناتها التي تم حفظها. يتيح المحرك هذه الكائنات في أثناء فتح قاعدة البيانات، يقوم المحرك بإتلاف كائنات Database وContainer وDocument.

ملاحظة قد ينتج عن أسماء كائنات Container بعض الاضطراب نظراً لتطابق ثلاثة منهم إلى أسماء ثلاثة كائنات فمجموعة أكسس. يحتوي كائن Container المسمى Forms للمحرك على معلومات إدارية عن جميع النماذج التي تم حفظها في قاعدة البيانات كما يحتوي كائن مجموعة أكسس المسمى Forms على كائنات Form التي تمثل النماذج المفتوحة الحالية "حتى تلك النماذج المفتوحة التي لم يتم حفظها بعد" كما يحتوي كائن Reports Container على المعلومات الإدارية الخاصة بكل التقارير المحفوظة، وتحتوي مجموعة Reports على كائنات Reports للتقارير المفتوحة. يحتوي كائن Container المسمى Modules للمحرك على معلومات إدارية عن جميع الوحدات النمطية التي تم حفظها في قاعدة البيانات بما في ذلك الوحدات النمطية القياسية ووحدات الفئة النمطية المرفقة بالنماذج والتقارير ووحدات الفئة النمطية المستقلة من النماذج والتقارير. يحتوي كائن مجموعة أكسس المسمى Modules على كائنات Module التي تمثل الوحدات النمطية المفتوحة حالياً.

يعرض إطار Database كائنات إطار Database التي تم حفظها. عند فتح وإغلاق هذه الكائنات، يقوم أكسس والمحرك بإنشاء وإتلاف الكائنات المؤقتة كما يلي:

- بالنسبة لكل جدول تم سرده في لوح Tables، يقوم Jet بتخزين كائن TableDef وكائن QueryDef لكل استعلام تم سرده، يستخدم Jet التعليمات التي تم تخزينها في كائن TableDef أو QueryDef لاسترجاع البيانات المخزونة وبإنشاء كائن Recordset وبعد ذلك يقوم أكسس بعرض طريقة عرض Database كتمثيل مرئي لكائن Recordset. وعند إغلاق الجدول أو استعلام، يقوم Jet بإتلاف مجموعة السجلات.
- بالنسبة لكل نموذج تم سرده في لوح Froms ولكل تقرير تم سرده في لوح Forms ولكل تقرير تم سرده في لوح Reports، يقوم أكسس بتخزين مجموعة مماثلة من التعليمات في ملف قاعدة البيانات. وعند فتح نموذج، يقوم أكسس بإنشاء كائن Form وإضافته إلى مجموعة Forms، بينما يقوم Jet بإنشاء وفتح أي مجموعة سجلات مضمنة مطلوبة من النموذج كمصدر سجل أو كمصادر بيانات لعناصر التحكم "مثل مصادر الصفوف لمربعات التحرير والسرد" يقوم أكسس بإنشاء وعرض النموذج بوضوح وعرض البيانات من مجموعة السجلات المتاحة في عناصر تحكم النموذج. وعند إغلاق النموذج، يتم إتلاف كائنات Form وrecordset المتطابقة. يحدث نفس التسلسل عند فتح تقرير.

- بالنسبة لكل وحدة نمطية قياسية وكل وحدة فئة نمطية مستقلة غير مرفقة بنموذج غير مرفقة بنموذج أو تقرير تم سرده في لوح Modules، يقوم أكسس بتخزين مجموعة مماثلة من التعليمات في ملف قاعدة البيانات. وعند فتح وحدة نمطية تم سردها في إطار Database، يقوم أكسس بعرض إطار Module وإنشاء كائن Module وإضافته إلى مجموعة Modules المطابق. إذا احتوى نموذج أو تقرير على وحدة نمطية، يمكنك فتح الوحدة النمطية المرفقة بشرط أن يكون النموذج أو التقرير مفتوحاً أيضاً. يقوم أكسس بعد ذلك بتشغيل فتح هذه الوحدات النمطية بنفس الطريقة التي يتم بها تشغيل وحدات الفئة النمطية القياسية والمستقلة وعند إغلاق الوحدة النمطية الخاصة بالنموذج أو التقرير أو عند إغلاق النموذج أو التقرير، يتم إتلاف كائن Module للوحدة النمطية المرفقة.
- يقوم أكسس، لكل ماكرو مدرج على لوحة Macros، بتخزين مجموعة مماثلة من الإرشادات في ملف قاعدة البيانات. عند فتح الماكرو، يقوم، أكسس بعرض إطار Macro، ولكنه لا يقوم بإنشاء كائن مماثل يمكن معالجته باستخدام إجراء VBA.

ولذلك، فإنك كما تعمل مع قاعدة بيانات مفتوحة، يقوم كل من أكسس وjet بإنشاء كائنات برمجة في الذاكرة التي تماثل أو تناظر الكائنات الطبيعية المعروضة على الشاشة، وأيضاً كائنات البرمجة الأخرى التي ليس لها عروض تقديمية مرئية. ويمكن استخدام إجراءات VBA في معالجة أي من كائنات البرمجة المتوفرة.

وباستخدام VBA يمكن أيضاً إنشاء كائنات جديدة "أو أمثلة" باستخدام أي من تصميمات أو تعريفات أنواع الكائنات "أو فئاتها" والتي يوفرها أكسس وJet. وهناك تقنيات خاصة بأكسس وJet لإنشاء أمثلة جديدة. وفيما يلي نوضح هذه التقنيات.

العناصر والخصائص الأساسية للبرمجة الموجهة عن طريق الكائنات

يتم في البرمجة الموجهة من جانب الكائنات، تجميع الكائنات المتشابهة داخل فئات وذلك على حسب التعريف. وتشترك كل الكائنات الموجودة في الفئة الواحدة في نفس الخصائص والأساليب. ويطلق على الكائن المحدد الذي ينتمي إلى الفئة "مثال" للفئة. وهناك أربعة عناصر أساسية لنظام البرمجة الموجهة عن طريق الكائنات، وهي:

التجريد: تشكيل نوع كائن يتضمن هذه الخصائص والأساليب المناسبة للغرض منه وتجاهل الجوانب الأخرى.

التجميع: تجميع الخصائص والأساليب كمكونات داخلية للكائن.

الوراثة: قدرة الكائنات الموجودة في الفئة التابعة على إعادة استخدام الخصائص

والأساليب الخاصة بالفئة الأصل بصورة تلقائية.

تعدد الأشكال: قدرة فئتان أو أكثر على أن يكون بهما أساليب تشترك في نفس الاسم والغرض منهما، ولكن لهما إرشادات مختلفة للتحقيق، مثل أساليب Requery الخاصة بكائنات DoCmd و Form و Control.

وبينما أن فيجوال بيسك ليست بصورة فعلية لغة برمجة موجهة من جانب الكائنات حيث أنها تنقصها الوراثة، فهي تقدم أغلب الميزات المطلوبة للتصميم الموجه عن طريق الكائنات، بما في ذلك القدرة على إنشاء تصميمات "وحدات نمطية للفئات" للكائنات الجديدة "كأمثلة للفئة". وفي إنشاء خصائص مخصصة "إجراءات خصائص وأساليب "أساليب عامة".

وفي VBA لأكسس، تعتبر الوحدات النمطية للنماذج والتقارير وحدات نمطية للفئة. ويمكن باستخدام أكسس ٢٠٠٠، إنشاء وحدات نمطية للفئات مستقلة عن النموذج أو التقرير.

إنشاء كائنات بيانات أكسس

يمكن القيام بالمهام التالية باستخدام VBA:

- إنشاء وتعديل قواعد بيانات وجداول وحقول وفهارس.
- إنشاء وتعديل علامات بين الجداول، وكذلك الجداول المرتبطة من قواعد البيانات الخارجية.
- إنشاء استعلامات جديدة لاسترداد البيانات من قاعدة البيانات ولإجراء تغييرات كبيرة بها.
- تعريف مستخدمين ومجموعات جدد للأغراض التأمينية.

وتستخدم أساليب ...Create لكائن الوصول إلى البيانات في إنشاء كائن تابع جديد، ويستخدم أسلوب OpenRecordset لإنشاء مجموعة سجلات. ويسرد الجدول ١٤-١ كائنات الوصول إلى البيانات والتي لها أساليب لإنشاء خصائص لنفسها وإنشاء كائنات تابعة.

الجدول ١٤-١: أساليب خاصة بإنشاء كائنات وخصائص الوصول إلى البيانات

الكائن	الأسلوب الخاص بإنشاء كائن آخر
DBEngine	CreateWorkspace
Workspace	CreateDatabase و CreateGroup و createUser.
Database	CreateProperty و createTableDef و createRelation و CreateQueryDef و openRecordset.
TableDef	CreateProperty و CreateField و createIndex و OpenRecordset.
QueryDef	CreateProperty و openRecordset.
Field	CreateProperty.
Index	CreateProperty و createField.
Relation	CreateField.
Recordset	OpenRecordset.
User	CreateGroup.
Group	CreateUser.

والخطوات العامة لإنشاء كائن بيانات أكسس هي:

١- استخدم أحد أساليب ...Create للكائن الأصل لإنشاء الكائن التابع.

٢- قم بتعريف سمات الكائن الجديد بإعداد الخصائص الخاصة به. وفي بعض الحالات ينبغي، قبل اكتمال الكائن، إنشاء كائنات تابعة له. فعلى سبيل المثال، عند إنشاء جدول ينبغي أيضاً إنشاء حقل واحد على الأقل وإلحاق الحقل بمجموعة Fields للجدول قبل أن يتم تعريف كائن TableDef. ويشبه ذلك إنشاء جدول بصورة تبادلية، ولن يسمح أكسس بحفظ جدول جديد ما لم يتم تعريف حقل واحد على الأقل.

٣- أضف كائن جديد إلى المجموعة المناظرة التي تنتمي إلى الأصل باستخدام أسلوب Append للمجموعة. ويمكن إلحاق كائن جديد فقط إذا كان مكتملاً.

ولإزالة كائن بيانات أكسس محفوظ "دائم" من قاعدة البيانات، فبصفة عامة يتم استخدام أسلوب Delete لحذف الكائن من مجموعته. ولإزالة كائن بيانات أكسس مؤقت "غير دائم" بما في ذلك كائنات Database و Workspace و Recordset، يتم تطبيق أسلوب Close للكائن لإغلاق الكائن بدلاً من حذفه.

ومن الضروري أن تتذكر أن كل كائن بيانات تتنوع خطوات الإنشاء الخاصة به أو خطوات إضافته إلى مجموعة وإزالته من قاعدة البيانات. ولقد تعرفت في الفصل ١١ على كيفية إنشاء وإزالة كائنات Recordset وفي الفصل ١٣ تعرفت على كيفية إنشاء وتخزين كائنات QueryDef. انظر الفصل ٦ لمزيد من المعلومات حول إنشاء أنواع كائنات بيانات أكسس الأخرى.

ولتوضيح الخطوات الخاصة بإنشاء كائن بيانات أكسس جديد في إجراء VBA، سنقوم الآن بإنشاء جدول وتكملته بفهرس وعلامة بجدول موجود.

استخدام التقنيات التنقلية في إنشاء جداول

هناك تقنيتان يمكن استخدامهما لإنشاء جدول، التقنية التنقلية وتقنية SQL. وسنتناول في هذا الجزء التقنيات التنقلية لإنشاء جدول جديد وإنشاء وإضافة فهرس وإنشاء علاقة بين جدولين في إحدى قواعد البيانات. وسنستخدم في ذلك أساليب ...Create متعددة.

وتشبه وسائط أسلوب ...Create خصائص الكائن الذي يتم إنشاؤه. وبينما تكون أغلب الوسائط خيارية في عبارة الأسلوب، فيمكن تجاهلها عند تنفيذ الأسلوب. إلا أن إلحاق الكائن الجديد بمجموعته يتطلب واحدة أو أكثر من الخصائص، فعلى سبيل المثال، لا يمكن إلحاق جدول جديد إذا لم يتم إعطاء الجدول اسم صالح. وإذا تم تجاهل وسيطة خيارية أو أكثر عند استخدام أسلوب ...Create، يمكن إعداد الخاصية المناظرة بواسطة عبارة تعيين قبل إلحاق الكائن بمجموعته. وبعد إلحاق كائن، تصبح العديد من الخصائص للقراءة فقط، ولا يمكن تغيير إعداداتها. "إذا كنت في حاجة بالفعل لتغيير خاصية للقراءة فقط لأحد الكائنات، ستحتاج إلى حذف الكائن وإنشاء آخر".

ملاحظة لإنشاء جداول باستخدام التقنيات التنقلية، ينبغي أن يكون العمل داخل مجال Jet، أي أن أكسس ينبغي أن يعمل مع أداة قاعدة بيانات Jet. وإذا كان العمل داخل نطاق ODBCDirect مع أكسس باستخدام أداة قاعدة بيانات أخرى، مثل SQL Server، فلن يمكن استخدام التقنيات التنقلية لإنشاء أو تعديل جداول داخل قاعدة البيانات. وفي الواقع ليس لنوع كائن ODBCDirect كائن TableDefs. إلا أنه يمكن استخدام أوامر لغة تعريف البيانات من SQL لإنشاء وتعديل جداول في أحد نطاقات ODBCDirect.

استخدام أسلوب CreateTableDef

يستخدم أسلوب CreateTableDef لإنشاء جدول داخل قاعدة البيانات أو إنشاء ارتباط بجدول داخل إحدى قواعد البيانات الخارجية. (يتم إنشاء ارتباط بجدول في إحدى قواعد البيانات الخارجية عن طريق إنشاء كائن TableDef لعرض الجدول المرتبط داخل قاعدة البيانات المستخدمة). والعبارة المستخدمة لإنشاء كائن TableDef جديد هي:

```
Set tdf = db.CreateTableDef(name, attributes, source, connect)
```

حيث يكون:

- tdf هو متغير من النوع TableDef الذي يعرض الجدول الجديد الذي يتم إنشاؤه.
- db هو مرجع لقاعدة بيانات مفتوحة يتم احتواء الجدول الجديد داخلها.
- Name هو متغير سلسلة خياري يقوم بتسمية الجدول الجديد.
- Attributes هو عدد صحيح طويل يعتبر مجموع القيم الثابتة الفعلية الخاصة بتحديد ميزات الجدول الجديد.
- Source هو الاسم الخياري للجدول الموجود في قاعدة بيانات خارجية ترغب في إنشاء ارتباط بها.
- Connect هي سلسلة خيارية تحتوي على معلومات حول نوع قاعدة البيانات والمسار لأحد الجداول المرتبطة ومعلومات يتم نقلها إلى ODBC وتحتوي برامج تشغيل قاعدة البيانات ISAM.

وبالرغم من أن كل الوسائط تعتبر خيارية، فإنه يتم إعداد خاصية Name لكائن TableDef قبل إلحاق الكائن. وينبغي أن يكون الاسم سلسلة فريدة في مجموعة TableDefs ويمكن أن يتكون مما لا يزيد عن ٦٤ حرف. فعلى سبيل المثال، تستخدم العبارة التالية لإنشاء جدول اسمه tblEmployeeExpenses داخل قاعدة البيانات الحالية:

```
Set tdf = CurrentDB.CreateTableDef("tblEmployeeExpenses")
```

ويمكن استخدام عبارتين أخرتين كبديل، وهما:

```
Set tdf = CurrentDB.CreateTableDef
tdf.Name = "tblEmployeeExpenses"
```

ملاحظة بعد إلحاق كائن TableDef جديد بمجموعة TableDefs تستأنف خاصية Name لتصبح خاصية للقراءة والكتابة. ويمكن التخلص من أو إزالة كائن TableDef باستخدام أسلوب Delete لمجموعة TableDefs.

استخدام أسلوب CreateField

ينبغي عند إنشاء جدول أن يتم إنشاء حقل واحد على الأقل له. ويستخدم أسلوب CreateField الخاص بالجدول لإضافة حقل له. والعبارة المستخدمة في ذلك هي:

```
Set fld = tdf.CreateField(name, type, size)
```

حيث يكون:

- Fld هو متغير كائن من النوع Field يعرض الحقل الذي يتم إنشاؤه.
- Tdf يشير إلى الجدول المحدد.
- Name هو متغير سلسلة خياري يعرف الحقل الجديد بصورة فريدة.
- Type هي ثابت أو قيمة ثابتة فعلية خيارية تقوم بتعريف نوع البيانات الخاص بالحقل الجديد.
- Size هو عدد صحيح خياري يحدد الحد الأقصى لكائن Field الذي يحتوي على النص.

وينبغي إعداد خاصيتي Name وtype قبل إلحاق حقل جديد إلى المجموعة. وكما هو الأمر مع أسماء الجداول، تتطلب خاصية Name سلسلة فريدة لا يزيد طول الأحرف بها عن ٦٤ حرف. "وعند تسمية الحقول تذكر أنه لا يمكن أن يحمل حقلان في مجموعة Fields للجدول نفس الاسم". ويتم إعداد خاصية Type إلى قيمة ثابتة حقيقية وصالحة "لا يمكن البحث عن "Type" في التعليمات الفورية للحصول على قائمة بالإعدادات الصالحة. وبالنسبة لنوع البيانات غير Text، يقوم إعداد خاصية Type بتحديد خاصية Size، لذلك فلن تكون هناك حاجة لتحديد حجم. وإذا كان نوع البيانات Text، يمكن إعداد خاصية Size على عدد صحيح أقل من ٢٥٥ أو تجاهل الإعداد لقبول الإعداد الافتراضي الخاص بقاعدة البيانات.

ملاحظة

بعد إلحاق كائن Field بمجموعة Fields الخاصة به، تستأنف خاصية Name لتصبح للقراءة والكتابة (فيما عدا تلك الخاصة بمجموعة Fields لأحد الجداول المرتبطة، إلا أن خاصية Type تصبح للقراءة فقط. ويمكن إزالة كائن Field من مجموعة Fields الخاصة به وذلك باستخدام أسلوب Delete للمجموعة. إلا أنه إذا تم تضمين الحقل داخل أحد الفهارس، فلن يمكن حذف الحقل إذا لم يتم أولاً حذف الفهرس.

إنشاء جدول

سنقوم الآن بإنشاء جدول جديد لتتبع حسابات الموظفين في قاعدة بيانـــات Northwind_Ch14.
افتح وحدة نمطية قياسية جديدة اسمها basNewTable وقم بإدخال إجراء NewTable الموضح فيما يلي:

```
Public Sub NewTable()
    Dim db As Database, tdf As TableDef
    Dim fld1 As Field, fld2 As Field, fld3 As Field, fld4 As Field
    Set db = CurrentDB
    Set tdf = db.CreateTableDef("tblEmployeeExpenses")
    Set fld1 = tdf.CreateField("ExpenseID", dbLong)
    fld1.Required = True
    ' To increment the value for new records
    fld1.Attributes = dbAutoIncrField
    Set fld2 = tdf.CreateField("EmployeeID", dbLong)
    fld2.Required = True
    Set fld3 = tdf.CreateField
    With fld3
        .Name = "ExpenseType"
        .Required = True
        .Type = dbText
        .Size = 30
    End With
    Set fld4 = tdf.CreateField("Amount", dbCurrency)
    With tdf.Fields
        .Append fld1
        .Append fld2
        .Append fld3
        .Append fld4
    End With
    db.TableDefs.Append tdf
    RefreshDatabaseWindow
End Sub
```

ويقـوم هـذا الإجـراء بإضافـة جـدول إلـى قـاعدة البيانـات الحاليـة اسـمه tblEmployeesExpenses. وهـو يقـوم بإنشـاء أربعـة حـقول، وهـي: ExpenseID

و EmployeeID و ExpenseType و Amount. يقوم بإعداد الخصائص الخاصة بها ثم إلحاق الحقول إلى مجموعة Fields للجدول الجديد. وبإتمام تعريف الجدول، يقوم هذا الإجراء بإلحاق الجدول الجديد بمجموعة TableDefs وتنشيط إطار Database.

قم بتنفيذ الإجراء NewTable داخل إطار Immediate. انقر داخل إطار Database ولاحظ الجدول الجديد. ويوضح الشكل ١٤-١ الجدول الجديد في عرض Design.

ملاحظة

لا يمكن إنشاء نوع بيانات AutoNumber بصورة مباشرة عند إنشاء حقل باستخدام أسلوب CreateField حيث أن Jet لا يمكنه التعرف على نوع بيانات AutoNumber. وبدلاً من ذلك يمكن إعداد نوع البيانات على dbLong ثم إعداد خاصية Attribute للحقل على القيمة الثابتة الفعلية dbAutoIncrField لزيادة قيمة الحقل للسجلات الجديدة آلياً. (يتم تطبيق إعداد dbAutoIncrField على جداول قاعدة بيانات mdb فقط).

ومن السهل ملاحظة أن أغلب خصائص الحقل التي يتم إعدادها في عرض Design للجدول، مثل Description و Caption و InputMask ليست خصائص مضمنة لكائن Field. وستتعرف على كيفية إعداد تلك الخصائص لأحد الحقول الجديدة لاحقاً في هذا الفصل تحت عنوان "إضافة خصائص معرفة من جانب التطبيق".

الشكل ١٤-١

يستخدم أسلوب CreateTable لإعداد جدول جديد، وأسلوب CreateField لإضافة حقل إلى الجدول الجديد.

إنشاء فهرس

يستخدم أسلوب CreateIndex للجدول المحدد لتعريف وتسمية فهرس جديد للجدول. والعبارة المستخدمة في ذلك هي:

```
Set idx = tdf.CreateIndex(name)
```

حيث يكون idx هو متغير كائن من نوع Index يعرض الفهرس الذي يتم إنشاؤه، ويشير tdf إلى الجدول، وnames هو متغير سلسلة خياري يقوم بتعريف الفهرس الجديد بصورة فريدة. وكما هو الأمر مع الجداول والحقول، يتم إعداد خاصية Name قبل إلحاق الفهرس الجديد وتحديد سلسلة فريدة لا تزيد في طولها عن ٦٤ حرف.



بعد إلحاق الفهرس الجديد، يمكن تغيير خاصية Name للجدول المحلي فقط، ولا يمكن تغيير خاصية Name لجدول مرتبط. ويمكن إزالة الفهرس من الجدول باستخدام أسلوب Delete لمجموعة Indexes.

وعند إنشاء فهرس جديد لجدول، ينبغي أيضاً إنشاء حقل واحد على الأقل للفهرس. ويستخدم أسلوب CreateField للفهرس المحدد الذي يتم إنشاؤه لإضافة حقل له. والعبارة المستخدمة في ذلك هي:

```
Set fld = idx.CreateField(name)
```

حيث fld هو متغير الكائن للنوع Field لحقل الفهرس الذي يتم إنشاؤه، بينما يشير idx إلى فهرس معين وname هو سلسلة تقوم بتعريف الحقل الجديد في الفهرس بصورة فريدة ولكنها تشير إلى حقل موجود في الجدول. وبالرغم من استخدام هذه العبارة في إنشاء حقل بالفهرس، فإنه لا يتم إضافة حقل جديد إلى الجدول، ولكن في الواقع تستخدم هذه العبارة في إنشاء حقل فهرس يقوم على حقل جدول موجود بالفعل. وبعد إنشاء حقول الفهرس، قم بإلحاقها بمجموعة Fields للفهرس الجديد ثم إلحاق الفهرس الجديد بمجموعة Indexes للجدول.

وسنقوم الآن، كمثال، بإنشاء فهرس أساس جديد يحتوي على حقل ExpenseID لجدول tblEmployeeExpenses الذي قمنا بإنشائه فيما سبق. ادخل إجراء NewIndex الموضح فيما يلي في وحدة basNewTable النمطية.

```
Public Sub NewIndex()
    Dim db As Database, tdf As Tabledef
    Dim idx As Index, fld As Field
    Set db = CurrentDB
```

```
    Set tdf = db.TableDefs("tblEmployeeExpenses")
    Set idx = tdf.CreateIndex("Primary")
    idx.Primary = True
    Set fld = idx.CreateField("ExpenseID")
    idx.Fields.Append fld
    tdf.Indexes.Append idx
End Sub
```

ويتم بهذا الإجراء إنشاء فهرس اسمه Primary لجدول tblEmployeeExpenses وإعداد خاصية Primary إلى True. ويستخدم هذا الإجراء أسلوب CreateIndex للفهرس في إضافة حقل فهرس جديد يقوم على حقل جدول EmployeeID الموجود بالفعل ويقوم بإلحاق حقل الفهرس بمجموعة Fields للفهرس ثم إلحاق الفهرس الجديد بمجموعة Indexes للجدول.

قم بتنفيذ إجراء NewIndex داخل إطار Immediate. وافتح جدول tblEmployeeExpenses في عرض Design ولاحظ أن ExpenseID هو المفتاح الأساسي الآن "انظر الشكل ١٤-٢". اغلق الجدول.

وتعتبر خصائص Primary وUnique وIgnoreNulls التي يتم إعدادها في عرض Design أيضاً خصائص لكائن Index، لذلك يمكن إعدادها في أحد إجراءات VBA. وبالإضافة إلى الخصائص المدرجة في عرض Design، يقوم Jet بتعريف الخصائص الأخرى لكائن Index "انظر الفصل ٦ لمزيد من المعلومات". فعلى سبيل المثال، يتوقف ترتيب السجلات التي تم إرجاعها على خاصية Attributes لكل حقل في الفهرس. ومن المفترض أن يكون حقل الفهرس قد تم فرزه على أساس ترتيب تصاعدي، إلا أنه يمكن إعداد خاصية Attributes إلى القيمة الثابتة الحقيقية dbDescending لفرز الحقل على أساس ترتيب تنازلي.

Index Name	Field Name	Sort Order
Primary	ExpenseID	Ascending

Indexes: tblEmployeeExpenses

Index Properties

Primary	Yes
Unique	Yes
Ignore Nulls	No

The name for this index. Each index can use up to 10 fields.

الشكل ١٤-٢

يستخدم أسلوب CreateIndex في إنشاء فهرس، بينما يستخدم أسلوب CreateField لإنشاء حقل فهرس يقوم على حقل جدول موجود.

إنشاء علاقات

يستخدم أسلوب CreateRelation لقاعدة البيانات لإنشاء وتسمية علاقة جديدة بين جدول أو استعلام أساسي وجدول أو استعلام خارجي، على التوالي. والعبارة المستخدمة هي:

```
Set rel = db.CreateRelation(name, table, foreigntable, attributes)
```

حيث:

- rel هو متغير كائن من نوع Relation يقوم بعرض العلاقة الجديدة.
- db يشير إلى قاعدة البيانات التي يتم فيها تعريف العلاقة الجديدة.
- Name هو متغير سلسلة خياري يقوم بصورة فريدة بتسمية العلاقة.
- Table هو متغير سلسلة خياري يقوم بتسمية الجدول أو الاستعلام الأساسي الموجود.
- Foreigntable هو متغير سلسلة خياري يقوم بتسمية الجدول أو الاستعلام الخارجي.
- Attributes هو متغير طويل خياري يحتوي على قيم ثابتة حقيقية لتحديد معلومات حول العلاقة.

قم بتضمين الوسائط بعبارة CreateRelation أو استخدام عبارات التعيين لإعداد الخصائص. وقم بإعداد الخصائص قبل إلحاق العلاقة، ولن يمكن تغيير أي من الخصائص بعد إلحاق العلاقة بمجموعة Relations لقاعدة البيانات.

وعند إنشاء كائن Relation جديد لعرض علاقة جديدة لجدولين أو استعلامين موجودين بالفعل، ينبغي أيضاً إنشاء حقل موجود في الجدول أو الاستعلام الأساسي، وينبغي تحديد اسم الحقل المطابق المناظر في الجدول الخارجي الموجود أو الاستعلام. ويستخدم أسلوب CreateField للعلاقة المحددة التي يتم إنشاؤها لإنشاء حقل جديد. والعبارة هي:

```
Set fld = rel.CreateField(name)
```

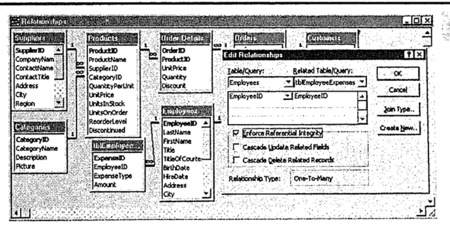
حيث fld هو متغير كائن من نوع Field يعرض حقل العلاقة، بينما يشير rel إلى العلاقة وname هو السلسلة التي تقوم بتعريف حقل العلاقة الجديد بصورة فريدة. وكما هو الأمر مع حقول الفهرس، لا تقوم هذه العبارة بإنشاء حقل جديد في الجدول، إلا أنها تقوم بتعريف حقل علاقة جديد لابد أن يقوم على حقل جدول موجود.

وكمثال على ذلك، وحيث أن الموظف الواحد قد يكون له سجلات حسابات عديدة، سنقوم بإنشاء علاقة بين جدول tblEmployeeExpenses الجديد "كجدول خارجي" وجدول Employees "كجدول أساسي". قم بإدخال إجراء NewRelation الموضح فيما يلي بوحدة basNewTable النمطية:

```
Public Sub NewRelation()
    Dim db As Database, rel As Relation, fld As Field
    Set db = CurrentDB
    Set rel = db.CreateRelation("ExpenseRelation")
    rel.Table = "Employees"
    rel.ForeignTable = "tblEmployeeExpenses"
    Set fld = rel.CreateField ("EmployeeID")
    fld.ForeignName = "EmployeeID"
    rel.Fields.Append fld
    db.Relations.Append rel
End Sub
```

ويتم بهذا الإجراء إنشاء علاقة جديدة اسمها ExpenseRelation. وذلك باستخدام جدول Employees كجدول أساسي وجدول tblEmployeeExpenses كجدول خارجي. ويتم بهذا الإجراء إنشاء حقل علاقة يقوم على حقل جدول EmployeeID في الجدول الأساسي وتحديد اسم الحقل المطابق في الجدول الخارجي "أيضاً EmployeeID". وبعد تعريف حقل العلاقة يقوم الإجراء بإلحاق حقل العلاقة الجديد بمجموعة Fields للعلاقة، ثم إلحاق العلاقة الجديدة بمجموعة Relations لقاعدة البيانات.

قم بتنفيذ الإجراء داخل إطار Immediate. اختر Tools ⇦ Relationships. وسيتم عرض تخطيط العلاقات. انقر زر ShowTable على شريط الأدوات، وحدد tblEmployeeExpenses ثم انقر Ok. ويقوم أكسس بإضافة جدول tblEmployeeExpenses إلى التخطيط وعرض العلاقة "انظر الشكل ١٤-٣".



الشكل ١٤-٣

يستخدم أسلوب CreateRelation لتعريف علاقة جديدة بين جدولين بينما يستخدم أسلوب CreateField لإضافة حقل علاقة يقوم على حقل موجود في الجدول الأساسي.

ملاحظـــة من المفترض أنه عند إنشاء علاقة بين جدولين في قاعدة البيانات الحاليـــة، يفرض أكسس التكامل المرجعي ولا يجعل التحديـــث أو الحـــذف متتاليـــاً، وينبغي تحديد هذه الإعدادات عـــن طريـــق خاصيـــة Attributes لكـــائن Relation قبل إلحاق العلاقة بمجموعة Relations. ويمكن البحـــث عـــن Attributes في التعليمات الفورية للحصول على قائمة بالقيم الثابتة الفعليـــة لكائن Relation.

استخدام تقنيات SQL في إنشاء جداول

عند استخدام تقنيات SQL في إنشاء الكائنات، يتم إنشاء استعلامات لتعريف البيانات يمكن تنفيذها بواسطة إجراءات VBA. وكما سبق توضيح ذلك في الفصل ١٣، إن استعلام تعريف البيانات هو أحد الاستعلامات المحددة من جانب SQL "تبدأ استعلامات تعريف البيانات بـــأحد أوامـــر SQL المدرجة في الجدول ١٣-٥، في الفصل ١٣". ويستخدم استعلام تعريف البيانات في إنشاء جداول وفهارس وتعديل جداول بواسطة إضافة أو إزالة فهرس، وتعريـــف علاقـــات وتـــأكيد التكـــامل المرجعي. ويمكن حفظ استعلام تعريف البيانات كاستعلام مخزن، أو يمكن تنفيـــذ عبـــارة SQL مباشرة باستخدام نفس التقنيات الخاصة بتنفيذ استعلام إجرائي أو عبارة SQL في أحد إجـــراءات VBA.

إنشاء جدول

العبارة الخاصة باستعلام تعريف البيانات والتي تستخدم لإنشاء جدول هي:

```
CREATE TABLE tablename (field1 type (size), field2 type (size), ...)
```

حيث:

- Tablename هو اسم الجدول الذي يتم إنشاؤه.
- Field1 وfield2 و... هي أسماء الحقول التي يتم إنشاؤها باستخدام العبارة "ينبغي إنشـــاء حقل واحد على الأقل".
- Type هو نوع البيانات للحقل الجديد.
- Size هو حجم الحقل بالأحرف "يتم تحديد الحجم فقط بالنسبة للنص والحقول الثنائية".

ويمكن أيضاً تضمين جملة CONSTRAINT لتعريف فهرس أو أكثر في نفس الوقت الذي يتم فيه تعريف الجدول الجديد. (جملة CONSTRAINT يتم توضيحها في الســـطور التاليـــة تحـــت عنوان "إنشاء علاقة").

ويتم عن طريق إجراء SQLNewTable، الموضح فيما يلي إنشاء جدول tblEmployeeExpenses. ويتم بهذا الإجراء تنفيذ عبارة SQL لاستعلام تعريف البيانات الذي يستخدم أمر CreateTable في إنشاء جدول يحتوي على أربعة حقول.

```
Public Sub SQLNewTable()
   Dim db As Database, strSQL As String
   strSQL = "Create Table tblEmployeeExpenses "
   strSQL = strSQL & "(ExpenseID COUNTER, EmployeeID LONG, _
      ExpenseType TEXT(30), Amount CURRENCY);"
   Set db = CurrentDB
   db.Execute strSQL
   RefreshDatabaseWindow
End Sub
```

ويحتوي SQL لأكسس على نوع بيانات COUNTER، لذلك يمكن إنشاء حقل AutoNumber مباشرة. (وعلى النقيض من ذلك، عند استخدام التقنيات التنقلية، يتم إنشاء حقل dbLong وإعداد خاصية Attributes له على dbAutoIncrField لزيادة قيمة الحقل لسجلات جديدة.)

تعديل حقل

يمكن استخدام أمر AlterTable لإضافة أو إفلات حقل واحد أو إضافة أو إفلات فهرس واحد من جدول موجود. وعبارة استعلام تعريف البيانات ALTER TABLE لإضافة أو إفلات حقل هي:

```
ALTER TABLE tablename {ADD COLUMN field type (size) |DROP
COLUMN field}
```

ووسائط tablename و field و type و size هي نفسها الخاصة باستعلام CREATE TABLE.

ولتعديل حقل، ينبغي أولاً حذفه ثم إضافة حقل جديد يحمل نفس الاسم. ويتم بإجراء SQLModifyTable تعديل حقل ExpenseType لزيادة حجم الحقل من ٣٠ إلى ٤٠ حرف.

```
Public Sub SQLModifyTable
   Dim db As Database, strDROP As String, strADD As String
   strDROP = "ALTER TABLE tblEmployeeExpenses "
   strDROP = strDROP & "DROP COLUMN ExpenseType;"
   strADD = "ALTER TABLE tblEmployeeExpenses "
```

```
    strADD = strADD & "ADD COLUMN ExpenseType TEXT (40);"
    Set db = CurrentDB
    db.Execute strDROP
    db.Execute strADD
End Sub
```

إنشاء فهرس

هناك ثلاث طرق لإنشاء فهرس لأحد الجداول هي:

- استخدام أمر CREATE TABLE عند إنشاء الجدول.
- استخدام أمر ALTER TABLE لإضافة حقل إلى جدول موجود.
- استخدام أمر CREATE INDEX لإضافة حقل إلى جدول موجود.

ولا تؤدي هذه الثلاث طرق إلى نفس النتائج. فإذا أردت تعريف علاقة وتأكيد تكامل مرجعي، ينبغي استخدام إما أمر CREATE TABLE أو أمر ALTER TABLE.

والعبارة الخاصة باستعلام تعريف البيانات CREATE INDEX هي:

```
CREATE [UNIQUE] INDEX indexname ON tablename (field1 [ASC|DESC], field2 [ASC | DESC], ...) [WITH {PRIMARY | DISALLOW NULL | IGNORE NULL}]
```

حيث:

- Indexname هو اسم الفهرس الذي يتم إنشاؤه.
- Field1 وfield2 و... هي أسماء الحقول التي يتم إنشاء الفهرس بها.
- تمنع كلمة UNIQUE الأساسية تكرار القيم في الحقل أو الحقول المفهرسة.
- تحدد كلمة PRIMARY الأساسية الحقل أو الحقول المفهرسة لتكون المفتاح الأساسي للجدول.
- يمنع خيار DISALLOWNULL إدخالات Null في الحقل أو الحقول المفهرسة للسجلات الجديدة.
- يمنع خيار IGNORNULL السجلات التي تحمل القيمة NULL في الحقل أو الحقول المفهرسة من أن يتم تضمينها داخل الفهرس. "تذكر أن الفهرس هو جدول منفصل يقوم Jet بإنشائه واستخدامه للبحث عن سجلات وذلك بإعداد هذا الخيار، ويعتبر الفهرس أصغر حجماً والبحث أكثر سرعة".

ويستخدم إجراء SQLCreateIndex. والموضح فيما يلي، أمر CREATE INDEX لإضافة فهرس إلى حقل ExpenseType ويحدد جملة WITH DISALLOW NULL ليطلب إدخال بالحقل.

```
Public Sub SQLCreateIndex()
    Dim db As Database, strIndex As String
    strIndex = "CREATE INDEX IndexExpenseType ON
tblEmployeeExpenses "
    strIndex = strIndex & "(ExpenseType) WITH DISALLOW NULL;"
    Set db = CurrentDB
    db.Execute strIndex
End Sub
```

استخدام استعلامات تعريف البيانات

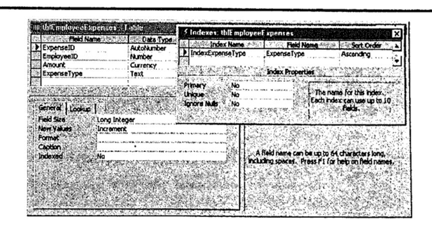
للتعرف على تقنيات SQL لإنشاء جدول، اتبع الخطوات التالية:

١- افتح إطار Relationships وقم بحذف العلاقة بين جدولي Employees وtblEmployeeExpenses. اغلق الإطار واحذف جدول tblEmployeeExpenses.

٢- ادرج إجراءات SQLNewTable وSQLModifyTable وSQLCreateIndex والسابق توضيحها وذلك في وحدة basNewTable النمطية.

٣- قم بتنفيذ الإجراءات داخل إطار Immediate بالترتيب الموضح في الخطوة الثانية.

٤- افتح جدول tblEmployeeExpenses الجديد في عرض Design. لاحظ تغير حجم حقل ExpenseType والفهرس الجديد "انظر الشكل ١٤-٤".



الشكل ١٤-٤

تستخدم استعلامات تعريف البيانات في إنشاء جدول أو تعديل حقل أو إنشاء فهرس.

إنشاء علاقة

يمكن إنشاء علاقة بين جدولين أو استعلامين باستخدام استعلامات تعريف البيانات التي تقوم على أمر CREATE Table أو ALTER TABLE. ويستخدم أمرا CREATE TABLE وALTER TABLE جملة CONSTRAINT لتعريف مفاتيح أساسية أو خارجية ولإنشاء علاقات وتأكيد تكامل مرجعي. وتعتبر CONSTRAINT قيد على القيم التي يمكن إدخالها بالحقل. وتستخدم جملة CONSTRAINT في عبارة CREATE TABLE أو ALTER TABLE في حالتين ، إنشاء أو حذف فهرس أو إنشاء أو حذف علاقة.

وهناك إصداران من جُمل CONSTRAINT، ويتوقف ذلك على ما إذا كنت تقوم بإنشاء قيد على حقل واحد أو قيد على أكثر من حقل. والعبارة الخاصة بجملة CONSTRAINT لإنشاء قيد على حقل واحد وإنشاء علاقة هي:

```
CONSTRAINT constraintname FOREIGN KEY (ref) REFERENCES
foreigntable foreignfield
```

حيث:

- Constraintname هو اسم القيد الذي يتم إنشاؤه.
- تحدد كلمة FOREIGN KEY، المستبدلة حسب الغرض أحد الحقول ليكون مفتاح خارجي.
- Ref هو اسم الحقل المطابق في الجدول الأساسي.
- Foreigntable هو اسم الجدول الخارجي.
- Foreignfield هو اسم الحقل المطابق في الجدول الخارجي.

فعلى سبيل المثال، جملة CONSTRAINT لوضع EmployeeID ليكون هو المفتاح الخارجي لجدول tblEmployeeExpenses وإنشاء علاقة مع جدول Employees هي:

```
CONSTRAINT IndexEmployee FOREIGN KEY (EmployeeID) REFERENCES
Employees (EmployeeID)
```

ويتم بإجراء SQLNewRelation. الموضح فيما يلي، إنشاء العلاقة بين جدولي Employees وtblEmployeeExpenses. ويضيف الإجراء علاقة إلى جدول tblEmployeeExpenses. مع تحديد جدول Employees ليكون هو الجدول الخارجي للعلاقة.

```
Public Sub SQLNewRelation()
    Dim db As Database, strRel As String
    strRel = "ALTER TABLE tblEmployeeExpenses "
    strRel = strRel & "ADD CONSTRAINT IndexEmployee "
```

```
    strRel = strRel & "FOREIGN KEY (EmployeeID) REFERENCES"
    strRel = strRel & "Employees (EmployeeID);"
    Set db = CurrentDB
    db.Execute strRel
End Sub
```

ولإجراء اختبار على هذا الإجراء، قم بإدراجه بوحدة basNewTable النمطية ثم قم بتنفيذه في إطار Immediate. افتح إطار Relationships ولاحظ العلاقة الجديدة.

الربط بالجداول الخارجية

يمكنك عن طريق العمل بصورة تبادلية، إجراء ربط بجدول موجود بقاعدة بيانات أخرى باختيار File ⇦ Get External Data ⇦ Link Tables. وبمجرد ربط الجدول الخارجي، يمكن الإشارة إليه بنفس الطريقة التي يشار بها إلى جدول محلي في قاعدة البيانات الحالية. فعلى سبيل المثال، إذا كان Expense Categories جدول مرتبط في قاعدة البيانات الحالية، يمكن إنشاء متغير كائن باستخدام عبارة التعيين:

```
Set tdf = CurrentDB.TableDefs("Expense Categories")
```

ويعتبر مفتاح إنشاء ارتباط بجدول خارجي باستخدام إجراء VBA هو خاصية Connect. وتستخدم العبارة التالية لإنشاء ارتباط بجدول موجود في قاعدة بيانات أخرى:

```
Set tdf = db.CreateTableDef(name, attributes, source, connect)
```

حيث:

- Name هو متغير سلسلة خياري. أي الاسم الذي تستخدمه في قاعدة البيانات الحالية لعرض الجدول المرتبط.
- Attributes هو عدد صحيح طويل خياري لتحديد سمات الجدول.
- Source هو تعبير السلسلة الخاص باسم جدول قاعدة البيانات الخارجية.
- Connect هو السلسلة الخيارية التي تحتوي على معلومات حول نوع قاعدة البيانات، ومسار الجدول المرتبط ومعلومات يتم نقلها إلى ODBC وبرامج تشغيل قواعد البيانات ISAM المحددة.

وتتضمن خاصية Connenct لكائن TableDef نوع قاعدة البيانات والمشار إلى قاعدة البيانات التي تحتوي على الجدول المرتبط. وإليك بعض الأمثلة لإعدادات الاتصال:

```
;DATABASE=c:\Program Files\Microsoft
Office\Office\Samples\northwind.mdb
Paradox 3.5;DATABASE=c:\Practice
FoxPro 2.6;DATABASE=c:\Practice
Excel 9.0;DATABASE=c:\Practice\data.xls
ODBC;DATABASE=mydatabase;UID=sn;PWD=eureka;DSN=tblCustomer
s
```

وبعد تحديد الاتصال بقاعدة البيانات كخاصية Connect، تستخدم خاصية SourceTableName لكائن TableDef لتحديد اسم جدول قاعدة البيانات الخارجية التي تريد الربط به.

ويتم بالإجراء LinkTable. الموضح فيما يلي، ربط جدول جديد في قاعدة البيانات الحالية بجدول Expense Categories في قاعدة البيانات Expenses.mdb "والتي تم إنشاؤها في الفصل ١".

```
Public Sub LinkTable()
   Dim db As Database, tdf As TableDef
   Set db = CurrentDB
   Set tdf = db.CreateTableDef("Expense Categories")
   tdf.Connect = ";DATABASE=c:\VBAHandbook\expenses.mdb"
   tdf.SourceTableName = "Expense Categories"
   db.TableDefs.Append tdf
   RefreshDatabaseWindow
End Sub
```

ويستخدم هذا الإجراء أسلوب CreateTableDef لإنشاء كائن TableDef في قاعدة البيانات الحالية. ويقوم الإجراء أيضاً بإعداد خصائص كائن TableDef الجديد لإنشاء الارتباط بجدول Expense Categories في قاعدة بيانات Expenses.mdb. "قد تحتاج إلى تغيير قاعدة البيانات Expenses.mdb على جهاز الكمبيوتر". ويحفظ الإجراء الارتباط بواسطة إلحاق كائن TableDef الجديد بمجموعة TableDefs لقاعدة البيانات الحالية.

ولإجراء اختبار على هذا الإجراء، قم بإدراجه داخل وحدة basNewTable النمطية وقم بتنفيذه داخل إطار Immediate. انقر داخل إطار Database. وبذلك يكون جدول Expense Categories قد تم ربطه. "وفي إطار Database، يسبق الجدول رمز على شكل سهم صغير إذا كان الجدول قد تم استيراده".

إنشاء نماذج وتقارير وعناصر تحكم

يقدم VBA لأكسس مجموعة من الوظائف لإنشاء نماذج وتقارير وعناصر تحكم وتفيد هذه الوظائف بصفة خاصة عند إنشاء معالج مخصص يقوم بإنشاء نموذج أو تقرير على حسب مواصفات المستخدم المجتمعة على شاشة المعالج. ويسرد الجدول ١٤-٢ وظائف أكسس المضمنة المستخدمة في إنشاء نماذج وتقارير وعناصر تحكم ومستويات لمجموعة تقارير والعبارات الخاصة بحذف عناصر التحكم. ويمكن اعتبار هذه الوظائف والعبارات أساليب لكائن Application.

الجدول ١٤-٢: وظائف وعبارات أكسس للعمل مع النماذج والتقارير

الوظيفة أو العبارة	الوصف
CreateForm وcreateReport	إنشاء نموذج أو تقرير، وفتح النموذج أو التقرير الجديد في حالة مصغرة في عرض Design وإرجاع كائن Report أو Form.
CreateControl وcreateReportControl	إنشاء عنصر تحكم على نموذج أو تقرير محدد وإرجاع كائن Control.
DeleteControl وdeleteReportControl	إزالة عنصر تحكم محدد من نموذج أو تقرير. وينبغي أن يكون النموذج أو التقرير مفتوح بشكل حالي في عرض Design.
CreateGroupLevel	إنشاء مجموعة أو فرز على حقل أو تعبير محدد في تقرير. وتقوم هذه الوظيفة بتجميع أو فرز البيانات وإنشاء رأس و/أو تذييل لمستوى المجموعة. وينبغي أن يكون التقرير مفتوحاً بصورة حالية في عرض Design.

وتستخدم وظيفة CreateForm بإنشاء نموذج مصغر في عرض Design باستخدام العبارة التالية:

```
Set frm = CreateForm(database, formtemplate)
```

حيث:

- Frm هو متغير كائن يعرض كائن النموذج الذي يتم إنشاؤه.

- ♦ Database هي سلسلة تقوم بتعريف اسم قاعدة البيانات التي تحتوي على قالب النموذج المستخدم.
- ♦ Formtemplate هو سلسلة تقوم بتعريف اسم النموذج المستخدم كقالب لتعريف النموذج الجديد.

وإذا تجاهلت الوسيطة الأولى، يتم استخدام قاعدة البيانات الحاليـة، وإذا تجـاهلت الوسـيطة الثانية، يقوم النموذج على القالب المحدد في تبويب Forms/Reports لمربع حـوار Options "يتاح ذلك باختيار Tools ⇦ Options".

والعبارة الخاصة بوظيفة CreateControl هي:

```
Set ctl = CreateControl(formname, controltype, section, parent,
columnname, left, top, width, height)
```

حيث:

- ♦ Ctl هو متغير كائن يعرض كائن Control الذي يتم إنشاؤه.
- ♦ Formname هو سلسلة تقوم بتعريف اسم النموذج المفتوح الذي تتم إضافة عنصر التحكم الجديد إليه.
- ♦ Controltype هو قيمة ثابتة حقيقية تقوم بتعريف نوع عنصر التحكم الجديد.
- ♦ Section هو قيمة ثابتة حقيقية تقوم بتعريف قسم النموذج الذي سوف يحتوي على عنصر التحكم الجديد.
- ♦ Parent هو سلسلة تقوم بتعريف اسم عنصر التحكم الأصل للعنوان أو خانة الاختيار أو زر الخيار أو عنصر تحكم مفتاح التبديل. "إذا كان عنصر التحكم الذي يتم إنشاؤه ليس له عنصر تحكم أصل، استخدم السلسلة ذات الطول الصفري لهذه الوسيطة".
- ♦ Columnname هو اسم الحقل الذي سينضم إليه عنصر التحكم. "إذا كنت تقوم بإنشاء عنصر تحكم غير منضم، استخدم السلسلة ذات الطول الصفري لهذه الوسيطة".
- ♦ Left وtop هما تعبيرات رقميان في تويب يشير إلى نظائر الزاوية اليسرى العليا لعنصر التحكم الذي يتم إنشاؤه.
- ♦ Width وheight هما تعبيران رقميان في تويب يشير إلى عرض وارتفاع عنصر التحكم.

والعبارة الخاصة بعبارة DeleteControl هي:

```
DeleteControl formname, controlname
```

وformname هو تعبير سلسلة يقوم بتعريف النموذج الذي يحتوي على عنصر التحكم
وControlname هو تعبير سلسلة يحدد عنصر التحكم الذي تريد حذفه.

ويستخدم إجراء NewForm. والموضح فيما يلي، وظيفتي CreateForm وCreateControl في إنشاء كائنات وعبارة DeleteControl لحذف أحد عناصر التحكم الجديدة:

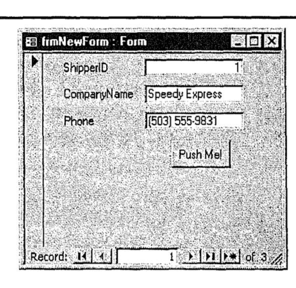
```
Public Sub NewForm()
   Dim db As Database, frm As Form
   Dim lbl As Label, txt As TextBox, cmd As CommandButton
   Dim Top As Integer, var As Variant
   Set db = CurrentDB
   Set frm = CreateForm
   With frm
      .RecordSource = "Shippers"
      .Width = 2.5
   End With
   Top = 100
   For Each var in db.TableDefs("Shippers").Fields
      Set lbl = CreateControl(frm.Name, acLabel, _
         acDetail, , , 300, Top, 1500, 230)
      Set txt = CreateControl(frm.Name, acTextBox, _
         acDetail, , , 1600, Top, 1500, 230)
      lbl.Caption = var.Name
      txt.ControlSource = var.Name
      Top = Top + 400
   Next
   Set cmd = CreateControl(frm.Name, acCommandButton, _
         acDetail, , , 2000, Top)
   cmd.Caption = "Push me!"
   cmd.Name = "cmdPush"
   cmd.SizeToFit
   ' Insert modification here
   With DoCmd
      .Restore
```

```
        .OpenForm frm.Name
        .RunCommand acCmdSizeToFitForm
    End With
    SendKeys "frmNewForm", False
    SendKeys "{Enter}", False
    RunCommand acCmdSaveAs
End Sub
```

ويتم بهذا الإجراء إنشاء نموذج جديد مصغر منضم إلى جدول Shippers وزيادة متغير Top إلى ١٠٠ توِيب" Top هي مسافة بين أعلى النموذج وأعلى العنوان التالي/مربع النص على أن يتم إنشاء اثنين".

وينتقل الإجراء عبر الحقول الموجودة في جدول Shippers. ويستخدم الإجراء لكل حقل في الجدول وظيفة CreateControl لإنشاء عناصر تحكم عنوان ومربع نص وإعداد خاصية Caption للعنوان وخاصية ControlSource لمربع النص إلى اسم الحقل. ويزيد الإجراء متغير Top بقدر ٤٠٠ توِيب ليتم بذلك لاثنين من العنوان التالي/مربع النص تعيين ٤٠٠ توِيب أسفل الزوج السابق. ويقوم الإجراء أيضاً بإنشاء زر أوامر. وبعد إنشاء عناصر التحكم يسترجع الإجراء النموذج، ويستخدم أسلوب OpenForm للانتقال إلى عرض Form ثم يقوم بتنفيذ الأمر المضمن لتغيير حجم النموذج. ويحفظ الإجراء النموذج باسم frmNewForm.

ولإجراء اختبار على إجراء NewForm. قم بإنشاء وحدة نمطية جديدة اسمها basNewForm وقم بإدراج الإجراء في الوحدة النمطية الجديدة. قم بتنفيذ الإجراء داخل إطار Immediate. ويوضح الشكل ١٤-٥ النموذج الجديد.



الشكل ١٤-٥

تستخدم وظيفة CreateForm وcreateComnt-rol في إنشاء نموذج في إجراء VBA.

تلميح عند تنفيذ أمر Save AS في أحد الإجراءات باستخدام عبارة RunCommand acCmdSaveAs. يقوم أكسس بعرض مربع Save As ويتوقف حتى تقوم بإدخال اسم الكائن الجديد وتنقر Ok أو تقوم بضغط Enter. ويمكن استخدام عبارة SendKeys لإرسال ضغط المفاتيح للاسم ولإغلاق مربع الحوار. وعند استخدام عبارة SendKeys قبل تنفيذ أمر Save As، يقوم أكسس بتخزين ضغط المفاتيح بعيداً حتى يحتاج إليها وذلك عن طريق مربع حوار Save As. قم بإعداد الوسيطة Wait على False لإرجاع عنصر تحكم إلى الإجراء بعد إرسال ضغط المفاتيح مباشرة.

إنشاء وحدات نمطية وإجراءات الأحداث

يوفر أكسس ٢٠٠٠ كائن Module مع بعض الخصائص والأساليب حتى يمكن تعريف وحدات نمطية جديدة وتعديل وحدات نمطية موجودة بصورة برمجية. ويمكن استخدام خاصية Module للنموذج في إنشاء وحدات نمطية. والعبارة المستخدمة في إنشاء وحدة نمطية وإرجاع مرجع إلى الوحدة النمطية الجديدة هي:

```
Set mdl = frm.Module
```

وmdl هو متغير كائن من نوع Module ويشير Form إلى النموذج.

ويمكن استخدام أسلوب CreateEventProc لكائن الوحدة النمطية لتعريف إجراء حدث جديد في وحدة نمطية موجودة. ويقوم أسلوب CreateEventProc بإنشاء قالب التعليمات البرمجية لإجراء حدث لحدث محدد، وأيضاً كائن محدد وإرجاع رقم السطر من السطر الأول في إجراء الحدث. وعبارة الأسلوب هي:

```
lngReturn = mdl.CreateEventProc(eventname, objectname)
```

حيث:

- LngReturn هو متغير يعرض رقم السطر من السطر الأول في إجراء الحدث.
- mdl هو متغير كائن من نوع Module.
- Eventname هو تعبير سلسلة يشير إلى اسم الحدث.
- Objectname هو تعبير سلسلة يحدد اسم الكائن.

وبعد إنشاء قالب التعليمات البرمجية، يستخدم أسلوب InsertLines لكائن Module لإدراج النص لإجراء الحدث. والعبارة المستخدمة في أسلوب InsertLines هي:

```
mdl.InsertLines line, string
```

حيث Line هو رقم السطر الذي يتم به بدء إدراج التعليمات البرمجية وstring هـو النـص الذي يتم إدراجه في الإجراء. ولإضافة عدة أسطر من التعليمات البرمجية، تستخدم القيمة الثابتـة الحقيقية vbCrLf لقطع السطر وبدء سطر جديد "يساوي ثابت vbCrLf الضغـط علـى مفتـاح Enter في نهاية سطر التعليمات البرمجية"، ويستخدم ثابت vbTab الحقيقي في إدراج تبويب.

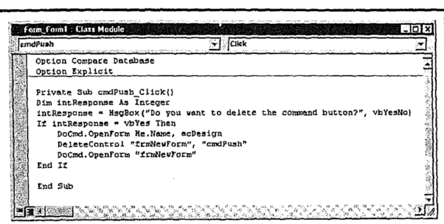
ولتوضيح إنشاء وحدة نمطية وإجراء حدث، سنقوم بتعديل إجراء NewForm الـذي سـبق توضيحه. ويعرض الإجراء المعدل رسالة للسؤال عما إذا كان يتم حذف الزر. فإذا نقر المستخدم Yes ينقل إجراء الحدث النموذج إلى عرض Design، ويحذف زر الأوامر ويعود مرة أخـرى إلى عرض Form.

اغلق نموذج frmNewForm. وحدد frmNewForm فـي إطـار Database واضغـط Delete لحذف النموذج. وبعد ذلك قم بإدراج السطور التالية من التعليمات البرمجية في إجـراء NewForm في السطر المميز بتعليق Insert modification here '.

```
Dim mdl As Module, lngReturn As Long, str As String
' Create a form module and return a reference to the module
Set mdl = frm.Module
' Create an event procedure for the command button's Click event
lngReturn = mdl.CreateEventProc("Click", "cmdPush")
str = "Dim intResponse as Integer" & vbCrLf
str = str & "intResponse = MsgBox(""Do you want to delete the "_
    & "command button?"",vbYesNo)" & vbCrLf
str = str & "If intResponse = vbYes Then" & vbCrLf
str = str & vbTab & "DoCmd.OpenForm Me.Name, acDesign" & vbCrLf
str = str & vbTab & "DeleteControl ""frmNewForm"", ""cmdPush""" _
    & vbCrLf
str = str & vbTab & "DoCmd.OpenForm ""frmNewForm""" &vbCrLf
str = str & "End If"
mdl.InsertLines lngReturn + 1, str
```

ويوضح التعديل أن المتغيرات في حاجة إلى تعليمات برمجية جديدة، ويقوم بإنشـاء وحـدة نمطية جديدة لنموذج، وإدراج قالب التعليمات البرمجية لإجراء الحدث cmdPush-Click الجديد. ويقوم الإجراء بإنشاء تعبير السلسلة لسطور إجراء الحدث. وتتطلب أسطر عديدة من التعليمـات البرمجية تضمين سلسلة داخل سلسلة، ويستخدم الإجراء اثنين من علامات التنصيص المزدوجـة لتمييز بداية ونهاية كل سلسلة داخل تعبير السلسلة. وبعد إنشاء متغير str الـذي يحتـوي علـى النص لإجراء الحدث، يستخدم الإجراء أسلوب InsertLines للوحدة النمطية لإدراج النص الذي يبدأ في السطر التالي للسطر الأول من قالب التعليمات البرمجية.

قم بتنفيذ إجراء NewForm المعدل في إطار Immediate، ويقوم الإجراء بإنشاء نموذج frmNewForm والوحدة النمطية للنموذج مع إجراء الحدث" انظر الشكل ١٤-٦". انقر زر Push Me في النموذج الجديد ثم انقر Yes. ويقوم الحدث بحذف زر الأوامر وعرض النموذج بدون زر الأوامر.



```
Form_Form1 : Class Module
cmdPush                     Click

Option Compare Database
Option Explicit

Private Sub cmdPush_Click()
Dim intResponse As Integer
intResponse = MsgBox("Do you want to delete the command button?", vbYesNo)
If intResponse = vbYes Then
    DoCmd.OpenForm Me.Name, acDesign
    DeleteControl "frmNewForm", "cmdPush"
    DoCmd.OpenForm "frmNewForm"
End If

End Sub
```

الشكل ١٤-٦

استخدم طريقة CreateEventPro c لإنشاء إجراء حدث وطريقة InsertLines لإدراج التعليمات البرمجية للاجراء.

حذف كائن إطار قاعدة بيانات

تستخدم طريقة DeleteObject لكائن DoCmd لحذف كائن إطار Database وبناء الجملة المستخدم هو:

DoCmd.DeleteObject objecttype, objectnam

يعتبر Objecttype ثابت فعلي اختياري يشير إلي نوع الكائن المراد حذفه ويعتبر objectname هو تعبير سلسلة اختياري يشير إلي اسم الكائن. لحذف الكائن تم اختياره في إطار Database ويمكن حذف كلا الوسيطتين.

علي سبيل المثال لحذف نموذج frmNewForm اغلق نموذج frmNewForm ثم ادخل DoCmd.DeleteObject acForm frmNewForm في إطار Im mediate ثم اضغط Enter.

إنشاء خصائص مخصصة

لكل كائنات تطبيق اكسس وكائنات وصول البيانات خصائص مضمنة تصف مميزاتهما ويكون لكل كائن مجموعة Properties تحتوي علي خصائصه المضمنة. تعلمت من خلال هذا الكتاب كيفية قراءة إعدادات الخصائص وكيفية إعداد قيم خصائص القراءة والكتابة وعلمت كذلك انه لا

يمكن تغيير أو حذف الخصائص المضمنة نفسها ولكن يمكن تغيير أو حذف قيمهم. العديد من طرق التعامل مع الكائنات التي تعلمتها تتضمن إعداد قيم الخصائص وهناك العديد من الحالات التي يكون من المناسب فيها تخصيص كائنات بواسطة إنشاء خصائص مخصصة.نقدم في الجزء التالي من هذا الفصل تقنيات إنشاء خصائص لكائنات وصول البيانات وكذلك نتعلم كيفية إنشاء خصائص جديدة لكائنات تطبيق اكسس.

إنشاء خصائص مخصصة لكائنات وصول البيانات

يمكن إنشاء خاصية مخصصة لكائن وصول البيانات وإضافة الخاصية المخصصة لمجموعة Properties الخاصة بالكائن وأبسط وسائل إنشاء خاصية مخصصة لكائن وصول البيانات تتضمن ثلاث خطوات:

١- استخدم طريقة CreateProperty لكائن وصول البيانات لإنشاء كائن Property باسم فريد.

٢- قم بإعداد خصائص Type وValue لكائن Property الجديد.

٣- استخدم طريقة Append لمجموعة Properties لإضافة الخاصية الجديدة لمجموعة Properties لكائن وصول البيانات.

بناء جملة طريقة CreateProperty هو:

```
set prp = object.CreateProperty(name, type, value, DDL)
```

حيث تكون:

- prp هو متغير كائن يمثل كائن Property الذي تقوم بإنشائه.
- Object هو مرجع للكائن الذي يتم إنشاء الخاصية من أجله.
- Name هو سلسلة اختيارية تطلق اسم فريد على الخاصية الجديدة.
- Type هو ثابت اختياري يعرف نوع البيانات للخاصية الجديدة.
- Value هو متغير اختياري يحتوي على القيمة المبدئية للخاصية.
- DDL هي إما True أو False وتشير إلي إذا ما كانت الخاصية كائن لغة تعريف بيانات (SQL).

استخدام خاصية معرفة بواسطة المستخدم

على سبيل المثال نقوم بإنشاء خاصية Tag لكل الأغراض لجدول Customers تحتوي معظم كائنات تطبيق أكسس على خاصية Tag الممكن استخدامها في تخزين المعلومات ولكن كائنات

وصول البيانات ليس لها خاصية Tag ويناقش جزء "استخدام خاصية Tag" تلك الخاصية بالنسبة للنماذج والتقارير وعناصر التحكم فيما يلي في هذا الفصل.

يجب عند إنشاء خاصية معرفة بواسطة المستخدم تحديد قيمة أولية قبل إضافتها إلى مجموعة Properties. يتم تحديد Type على dbText ليحتفظ بقيم النص. بالنسبة للنموذج أو تقرير أو عنصر تحكم فأن خاصية Tag المضمنة لها سلسلة طولها صفر ("") كقيمة افتراضية ومع ذلك لا يمكن استخدام سلسلة طولها صفر لخاصية معرفة بواسطة المستخدم ولذلك نقوم بإعداد القيمة الأولية على "My tag".

يقوم إجراء TableTag بإنشاء خاصية Tag:

```
Public Sub TableTag()
   Dim db As Database, tdf as TableDef, prp As Property
   On Error GoTo Error_TableTag
   Set db = CurrentDB
   Set tdf = db.TableDefs("Customers")
   Set prp = tdf.CreateProperty("Tag")
   prp.Type = dbText
   prp.Value = "My tag"
   tdf.Properties.Append prp
Exit_TableTag:
   Exit Sub
Error_TableTag:
   If Err = 3367 Then
   ' The property has already been appended to the collection.
      Resume Next
   Else
      MsgBox Err.Number & Err.Description
      Resume Exit_TableTag
   End If
End Sub
```

نقوم في هذا الأجراء بتضمين خطأ معالجتنا لأنه إذا حاولت إرفاق خاصية تم إرفاقها من قبل يظهر خطأ وقت التشغيل "كود ٣٣٦٧" ويكتشف الأجراء هذا الخطأ فإذا كانت الخاصية قد تم إرفاقها فأن العبارة التي تشغل طريقة Append تقوم بإظهار الخطأ وتتيح عبارة Resume Next للإجراء الاستمرار بدون تنفيذ طريقة Append وإذا وقع خطأ مع كود خطأ آخر يعوض مربع الرسالة معلومات الخطأ وينتهي الأجراء.

ملحوظة يجب إرفاق خاصية معرفة بواسطة المستخدم إلي مجموعة الخصائص لكائن وصول البيانات قبل حفظ الخاصية في قاعدة البيانات وإذا انتهي الأجراء بدون إرفاق الخاصية يتم التخلص من الخاصية.

لاختيار جدول TableTag قم بإنشاء وحدة نمطية جديدة تسمى basProperties وادخل الأجراء ثم تأكد من إغلاق جدول Customers وقم بتشغيل إجراء TableTag في إطار Immediate. عند تشغيل إجراء TableTag يتم إنشاء خاصية Tag وتحفظ في قرص وتصبح خاصية دائمة في الجدول أما بالنسبة للخصائص المعرفة بواسطة المستخدم فهي لا تظهر في أوراق الخصائص ولذلك لا يمكن إعدادهم بصورة تفاعلية.

الإشارة إلي الخاصية المعرفة بواسطة المستخدم

عند الإشارة إلي خاصية معرفة بواسطة المستخدم في إجراء يجب تضمين مرجع صريح لمجموعة Properties ويجب استخدام اسم الخاصية في العبارة وفي تلك الحالة يمكن استخدام إما عبارة نقطة علامة التعجب أو عبارة الأقواس كما يلي:

```
object.Properties!propertyname
object.Properties("propertyname")
```

ملحوظة لا يمكن الإشارة إلي خاصية معرفة بواسطة المستخدم استخدام عبارة object.propertyname المختصرة أو عبارة المرجع بواسطة رقم الفهرس object.Properties(n).

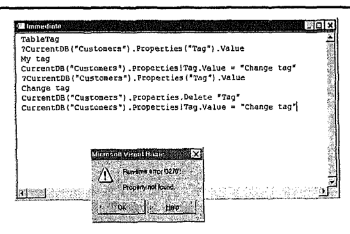
على سبيل المثال للإشارة إلي خاصية Tag لجدول Customers قم بإدخال ما يلي في إطار Immediate:

- Type ?CurrentDB("Customers").Properties("Tag").Value واضغط Enter. يكتب إطار Immediate كلمة My tag.
- Type CurrentDB("Customers").Properties!Tag.Value = "Change واضغط Enter. تتغير قيمة خاصية Tag المخصصة.
- Type ?CurrentDB("Customers").Properties("Tag").Value واضغط Enter يكتب إطار Immediate كلمة change tag.

حذف خاصية معرفة بواسطة المستخدم

يمكن حذف خاصية معرفة بواسطة المستخدم بتشغيل طريقة Delete لمجموعة Properties لكائن وصول البيانات. على سبيل المثال حذف خاصية Tag المخصصة لجدول Customers عن طريق كتابة CurrentDB ("Customers"). Properties. Delete "Tag" في إطار Immediate والضغط على Enter وبذلك يتم حذف خاصية Tag المخصصة.

لتأكيد الإلغاء اكتب CurrentDB ("Customers").Properties!Tag.Value="Change tag" واضغط Enter فيظهر الخطأ لأن خاصية Tag لم تعد موجودة، "راجع الشكل ١٤-٧".



الشكل ١٤-٧ رسالة الخطأ التي تظهر عند الإشارة إلي خاصية معرفة بواسطة المستخدم لم تعد موجودة.

إضافة الخصائص المُعرفة بواسطة التطبيق

نوع خاص من الخصائص المعرفة بواسطة المستخدم. كما أوضحنا في الفصل ٦ عندما يقوم تطبيق مثل أكسس بتعريف خاصية لكائن وصول بيانات تعرف الخاصية على أنها خاصية معرفة بواسطة التطبيق. لا يتعرف محرك Jet بصورة افتراضية بالخصائص المعرفة بواسطة التطبيق ويدرج الجدول ١٤-٣ أمثلة لخصائص معرفة بواسطة التطبيق قام أكسس بتعريفها.

الجدول ١٤-٣: خصائص المُعرفة بواسطة التطبيق لأكسس

الكائن	الخصائص المعرفة بواسطة اكسس
Database	AppTitle, StartupShowDBWindow, StartupShowStatusBar, AllowShortcutMenus, AllowFullMenus, AllowBuiltInToolbars, AllowToolbarChanges, AllowBreakIntoCode, AllowSpecialKeys, Replicable, ReplicationConflictFunction

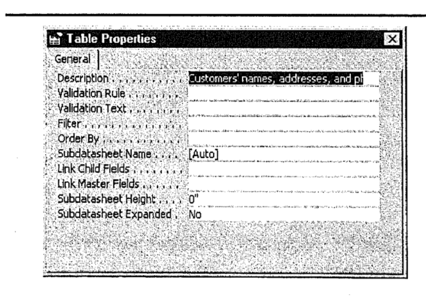
الجدول ١٤-٣: خصائص المُعرفة بواسطة التطبيق لأكسس

الكائن	الخصائص المعرفة بواسطة اكسس
TableDef	DatasheetFontHeight, DatasheetFontName, Description, FrozenColumns, RowHeight, ShowGrid, KeepLocal
QueryDef	DatasheetFontHeight, DatasheetFontName, LogMessages, RecordLocks, RowHeight, ShowGrid, Description, KeepLocal, UseTransaction
Field	Caption, DecimalPlaces, ColumnHidden, Description, ColumnOrder, Format, ColumnWidth, InputMask

هناك طريقتين ليستطيع Jet التعرف على خاصية معرفة بواسطة التطبيق:

- إعداد قيمة الخاصية لأول مرة في واجهة المستخدم.
- إعداد قيمة الخاصية لأول مرة في تعليمات VBA البرمجية.

إعداد خاصية معرفة بواسطة التطبيق في الواجهة عند إنشاء جدول بصورة تفاعلية يمكن إعداد خاصية Description في طريقة عرض Design الخاصة بالجدول. إذا أدخلت قيمة لخاصية Description في ورقة الخاصية يضيف Jet بصورة آلية خاصية Description لمجموعة Properties الخاصة بالجدول ويوضح الشكل ١٤-٨ ورقة خاصية Table Properties لجدول Customers مع إعداد أولي لخاصية Description الخاصة بالجدول.



الشكل ١٤-٨

عند تعيين خاصية معرفة بواسطة التطبيق في ورقة خصائص كائن يضيف Jet الخاصية لمجموعة Properties الخاصة بالكائن.

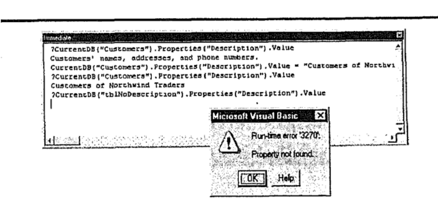
اتبع الخطوات التالية لاستكشاف كيف يعالج Jet خاصية Description عند إدخال قيمة بصورة تفاعلية أو عند عدم إدخال قيمة:

١- في إطار Immediate ادخل ?CurrentDB ("Customers").Properties ("Description")Value واضغط Enter فيتم عرض الإعداد الحالي لخاصية Description.

٢- ادخل CurrentDB("Customers").Properties("Description").Value= "Customers of Northwind Traders" واضغط Enter ثم ادخل ?CurrentDB("Customers").Properties("Description").Value ثم اضغط Enter ويعرض إطار Immediate القيمة الحالية للخاصية.

٣- افتح جدول Customers في طريقة عرض التصميم واعرض مربع حوار Table Properties فتعرض خاصية Description الإعداد الذي تم تغييره.

٤- أنشئ جدول جديد يسمى tb1NoDescription مع وجود حقل CustomerName منفرد في إطار Immediate ادخل ?CurrentDB("tb1No Description").Properties ("Description").Value ثم اضغط Enter ولا يستطيع Jet التعرف على خاصية Description للجدول الجديد ويعرض رسالة خطأ، راجع الشكل ١٤-٩.



الشكل ١٤-٩
رسالة الخطأ التي تظهر عندما لا يتعرف Jet على خاصية معرفة بواسطة التطبيق.

إعداد خاصية معرفة بواسطة التطبيق باستخدام التعليمات البرمجية إذا أردت إعداد خاصية معرفة بواسطة التطبيق لكائن لأول مرة باستخدام التعليمات البرمجية. يجب أولا لإنشاء الخاصية وإرفاقها لمجموعة Properties وعلى سبيل المثال ينشئ إجراء ApplicationProperty الموضح فيما بعد خاصية Description لجدول tb1NoDescription.

```
Public Sub ApplicationProperty()
   Dim db As Database, tdf As TableDef, prp As Property
```

```
    Set db = CurrentDB
    Set tdf = db.TableDefs("tblNoDescription")
    Set prp = tdf.CreateProperty("Description")
    prp.Type = dbText
    prp.Value = "There is a Description"
    tdf.Properties.Append prp
    Debug.Print tdf.Properties!Description
End Sub
```

لاختيار إجراء ApplicationProperty يتم إدخاله في الوحدة النمطية basProperties ويتم تشغيل الإجراء في إطار Immediate حيث ينشئ الإجراء خاصية Description لجدول tb1NoDescription ثم يعرض الإعداد الجديد للخاصية.

إعداد خيارات بدء التشغيل برمجيا: من أهم الخصائص المعرفة بواسطة التطبيق والتي لا يمكن إعدادها في واجهة المستخدم ويجب إعدادها لأول مرة باستخدام تعليمات VBA البرمجية هي خاصية AllowBypassKey لقاعدة البيانات. تتيح خاصية AllowBypassKey تحديد إذا كان المستخدم يستطيع تجاوز خيارات بدء التشغيل وماكرو بدء التشغيل AutoExec التي قمت بإعداده للتطبيق. عندما تكون قيمة AllowBypassKey هي True أي القيمة الافتراضية يؤدي الضغط على مفتاح Shift عند بدء التشغيل إلي التجاوز عن شروط بدء التشغيل.عندما تكون قيمة AllowBypassKey هي False يصبح تأثير الضغط على مفتاح Shift عند بدء التشغيل بغير فائدة ولا تكون هذه الخاصية متاحة في مربع Startup.

يمكن استخدام إجراء SetByPass لإعداد القيمة لخاصية AllowBypassKey

```
Public Sub SetByPass(booByPass As Boolean)
    Dim db As Database, prp As Property
    On Error GoTo Error_SetByPass
    Set db = CurrentDB
    db.Properties!AllowBypassKey = booByPass
Exit_SetByPass:
    Exit Sub
Error_SetByPass:
    If Err = 3270 Then
    ' The property does not exist and needs to be created
        Set prp = db.CreateProperty("AllowBypassKey")
        prp.Type = dbBoolean
```

```
            prp.Value = booByPass
            db.Properties.Append prp
            Resume Next
        Else
            MsgBox Err.Number & Err.Description
            Resume Exit_SetByPass
        End If
    End Sub
```

يستخدم هذا الأجراء معالجة الأخطاء لتحديد إذا ما كانت الخاصية قد تم إرفاقها لمجموعة Properties ويعتبر هذا الأجراء مثال على إيجاد خطأ بطريقة متعمدة كجزء من منطق الأجراء.

يبدأ أجراء SetByPass بمحاولة إعداد خاصية AllowBypassKey إذا كانت الخاصية قد تم إضافتها إلي قاعدة البيانات ويتم تنفيذ العبارة المحددة لتعيين قيمة الخاصية وينتهي الأجراء.إذا كانت الخاصية لم تتم إضافتها على قاعدة البيانات تفشل عبارة التعيين ويظهر الخطأ "كود ٣٢٧٠٩" ويتولى الأمر كود معالجة الأخطاء. يقوم كود معالجة الأخطاء بإنشاء الخاصية وإعداد قيمتها ويرفق الخاصية بمجموعة Properties في قاعدة البيانات.

لاستخدام الأجراء استدعي الأجراء وحدد الوسيطة على False إذا أردت عدم إتاحة مفتاح Shift كأحد شروط التجاوز عند بدء التشغيل أو حدد الوسيطة على True إذا أردت إتاحة مفتاح Shift كأحد شروط التجاوز ثم اتبع الخطوات التالية لاختبار الأجراء:

١- ادخل أجراء SetBypass في الوحدة النمطية basProperties.

٢- قم بالتبديل إلي إطار Database واختر Tools ⇦ Startup وقم بإعداد Application Title على Testing the Bypass ثم انقر Ok فيعرض شريط العنوان الجديد.

٣- لإلغاء إتاحة تأثير التجاوز من مفتاح Shift عند بدء التشغيل ادخل SetByPass (False) في إطار Immediate واضغط Enter بذلك تكون خاصية AllowBypassKey قد أضيفت على قاعدة البيانات وتم إلغاء تأثير مفتاح Shift عند بدء التشغيل.

٤- اغلق قاعدة البيانات اضغط مفتاح Shift ثم ابدأ تشغيل قاعدة البيانات يستمر شريط العنوان في عرض Testing the Bypass قم بإتاحة تأثير التجاوز لمفتاح Shift عند بدء التشغيل بإدخال SetBypass (True) في إطار Immediate ثم الضغط على Enter تلك المرة عند إغلاق قاعدة البيانات والضغط على مفتاح Shift في أثناء بدء تشغيل قاعدة البيانات ويتم إلغاء إتاحة خيارات بدء التشغيل ويعرض شريط العنوان Microsoft Access.

إنشاء خصائص مخصصة للنماذج والتقارير وعناصر التحكم

لا توجد طريقة لإضافة خصائص مخصصة لمجموعة Properties لنموذج أو تقرير أو عنصر تحكم ومع ذلك هناك طريقتين يمكن بهما إنشاء خصائص مخصصة لتلك الكائنات من تطبيقات أكسس وأبسط وسائل إنشاء خاصية مخصصة لنموذج أو تقرير أو عنصر تحكم هي استخدام إجراءات الخصائص ويصف الجزء التالي التقنيتين.

استخدام خاصية Tag

لكل من النماذج أو مقاطع النماذج والتقارير أو مقاطع التقارير وعناصر التحكم خاصية Tag التي يمكن استخدامها لتخزين تعبير سلسلة.الإعداد الافتراضي هو سلسلة طولها صفر (" ") ولكن يمكن تخزين أي تعبير سلسلة يصل إلي ٢٠٤٨ حرف ويمكن إعداد خاصية Tag في ورقة الخصائص أو في إجراء VBA.

استخدمت خاصية Tag في الفصل ٩ لمتابعة كيفية فتح نموذج وإذا تم فتح النموذج بالنقر على زر يوجد على نموذج غير نموذج Main Switchboard يتم تخزين اسم النموذج مع زر الفتح. استخدمت الإجراءات المستخدمة لإغلاق النموذج خاصية Tag لتحديد أي النماذج لا يكون مخبأ. في تلك الأمثلة تم استخدام خاصية Tag كخاصية Formopener مخصصة.

يمكن كذلك تخزين خصائص مخصصة متعددة في خاصية Tag وتحتاج فقط إلي طريقة لتعريف القيم وإحدى الهياكل المفيدة لها بناء الجملة التالي:

```
TagName1=TagValue1;TagName2=TagValue2;...;TagNameN=TagValu-
eN
```

يتم في بناء هذه الجملة فصل الأسماء والقيم بواسطة علامة يساوي(=) ويتم فصل الأزواج بواسطة فاصلة منقوطة (;) في أي قيم.لاستخدام بناء جملة مثل هذا تحتاج إلي إنشاء إجراءات للتعامل تعتمد على السلسلة والتي تستطيع قراءة إعداد خاصية Tag ثم ابحث عن اسم خاصية مخصص محدد وأقرأ القيمة المناظرة له.

ملحوظة البناء الخاص بتخزين خصائص مخصصة متعددة في خاصية Tag الموضح هنا يقدمه كتاب Access2000 Developer'sHandbook للكتاب PaulL Itwin وKEN Getz وmike Gilbert لدار نشر Sybex لعام ١٩٩٩ وللحصول على مزيد من المعلومات راجع هذا الكتاب.

إنشاء إجراءات خاصية

قام الفصل ٧ بتقديم ثلاث أنواع من الإجراءات وهم إجراءات دالة والإجراءات الفرعية وكذلك إجراءات الخصائص. تستخدم إجراءات الخاصية لإنشاء خصائص مخصصة للنماذج والتقارير والكائنات الجديدة التي تنشئها باستخدام وحدات نمطية ذات فئة مستقلة.

تأتي إجراءات الخاصية عادة بطريقة زوجية حيث ينشئ إجراء Property Let خاصية يمكن تعيين قيمتها وينشئ إجراء Property Get خاصية يمكن قراءة قيمتها.

ملحوظة عند إنشاء خاصية مخصصة يتم إعدادها بالإشارة إلي كائن بدلا من تعيين قيمة ولذلك استخدم إجراء Property Set لإنشاء الإجراء الذي قمت بإعداده واستخدام إجراء Property Get لإنشاء إجراء يقوم بإرجاع مرجع لكائن. تعتبر تلك الخصائص المخصصة مشابهة لخصائص النموذج والتقرير والوحدة النمطية وكذلك Recordsetclone التي تقوم بإرجاع كائنات بدلا من قيم.

إنشاء إجراء Property Let: لإنشاء خاصية مخصصة لنموذج أو تقرير تستخدم عبارة Property Let لإنشاء إجراء خاصية في الوحدة النمطية للنموذج أو التقرير ويكون بناء الجملة في أبسط الحالات هو:

```
Public Property Let propertyname (propertyvalue As datatype)
    [statements]
End Property
```

يعتبر Propertyname هو اسم الخاصية المخصصة المراد إنشائها وتمثل Propertyvalue قيمة الخاصية المخصصة. عند تعريف الكائن باستخدام وحدة نمطية ذات فئة مستقلة يتم إنشاء إجراء Property Let في الوحدة النمطية.

يتم تشغيل إجراء Property Let عن طريق تضمين اسم الإجراء كخاصية في عبارة تعيين تحدد قيمة للخاصية. على سبيل المثال يعمل إجراء LockedFormPropertyLet كلما تم إعداد القيمة باستخدام عبارة تعيين مثل:

```
Forms!Suppliers.LockedForm = True
```

عادة يتم إنشاء إجراء خاصية كإجراء عام يخزن في الوحدة النمطية للنموذج. إذا أردت أن تعمل خاصية مخصصة مثل خاصية نموذج مضمنة يجب أن تكون عامة ليتم استدعائها في وحدات نمطية أخرى.

ملحوظة لا تظهر خاصية مخصصة لنموذج في ورقة خصائص النمــوذج لذلــك لا يمكن إعداد خاصية مخصصة في وضع التصميــم ولكــن يمكــن إعــداد الخاصية في إجراء VBA إذا كان إجراء عام.

على سبيل المثال نقوم بإنشاء LockedForm كخاصية مخصصة لنمــوذج Suppliers يتــم إنشاء الخاصية لأنه عند إعداد قيمة خاصية LockedForm على True يتم تأمين عناصر التحكم الموجودة على النموذج وعند إعداد قيمة خاصية LockedForm على False تصبــح عنــاصر تحكم البيانات الموجودة على النموذج غير مؤمنة وهذا يعني أن الإجراء يؤمن عنــاصر تحكــم مربعات النص فقط ولكن يمكن تعديل الإجراء لتأمين أو إلغاء تأمين أي نوع من عناصر تحكــم البيانات وما يلي يوضح إجراء Property Let LockedForm.

```
Public Property Let LockedForm(booLock As Boolean)
   Dim ctl As Control
   Select Case booLock
   Case True
      If Me.Dirty = True Then
         RunCommand acCmdSaveRecord
      End If
      For Each ctl In Me
         If TypeOf ctl Is TextBox Then ctl.Locked = True
      Next
   Case False
      For Each ctl In Me
         If TypeOf ctl Is TextBox Then ctl.Locked = False
      Next
   End Select
End Property
```

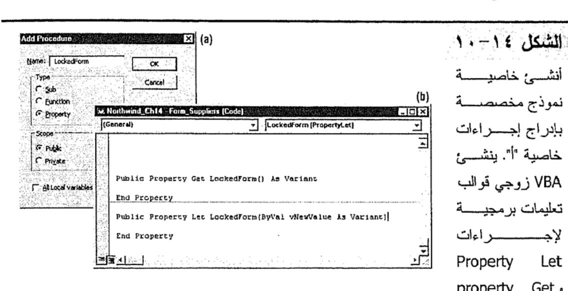
يتم تمرير قيمة الخاصية كوسيطة للإجراء ويحدد الإجراء إذا كانت القيمة True أو False فــي أي الحالات يستخدم الإجراء بناء Next For Each للدوران حول عنــاصر التحكــم علــى النموذج.يمكن الإشارة إلي مجموعة Controls علــى أنــها Me.Controls أو أنــها Me لان مجموعة Controls هي المجموعة الافتراضية للنموذج. عند إعداد خاصية LockedForm على True يتم اختبار إجراء الخاصية أولا لتحديد إذا ما كانت هناك تغييرات لم يتــم حفظــها علــى السجلات ثم تحفظ تلك التغييرات إن وجدت ثم تؤمن كل عناصر تحكم مربع الحوار وعند إعــداد خاصية LockedForm على False يقوم الأجراء بإلغاء تأمين عناصر التحكم.

اتبع الخطوات التالية لاختبار الأجراء:

١- حدد نموذج Suppliers في إطار Database وانقر زر Code في شريط الأدوات.

٢- اختر Insert>Procedure ادخل LockedForm في حقل Name وحدد خيار Property في مربع حوار Add Procedure راجع الشكل ١٤-١٠أ حيث يدرج VBA آليا قوالب تعليمات برمجية لكل من إجراءات Property Let وProperty Get.

٣- ادرج إجراء Property Let LockedForm.



الشكل ١٤-١٠

أنشئ خاصية نموذج مخصصة بإدراج إجراءات خاصية "أ". ينشئ VBA زوجي قوالب تعليمات برمجية لإجراءات Property Let وproperty Get الجديدة "ب".

١- في قالب التعليمات البرمجية لإجراء Property Get LockedForm يتم تغيير نوع بيانات قيمة إرجاع الإجراء من Variant "متباين" إلي Boolean "منطقي". يقوم إجراء Property Get في الجزء التالي ولكن يجب، في الوقت الحالي، أن تتطابق أنواع البيانات لتفادي الوقوع في أخطاء في وقت التشغيل.

٢- احفظ الوحدة النمطية وقم بالتبديل إلي طريقة عرض النموذج. قم بتشغيل الإجراء بكتابة Forms!Suppliers.LockedForm=TrueForms!Suppliers.LockedForm=Tr-ue في إطار Immediate ثم اضغط Enter ثم تأكد أن عناصر تحكم البيانات قد تم تأمينها.

٣- ادخل Forms!Suppliers.LockedForm=False ثم اضغط Enter وأكد أن عناصر تحكم البيانات غير مؤمنة.

إنشاء إجراء Property Get عند إنشاء خاصية مخصصة للكتابة فقط تسمى Propertyname باستخدام إجراء Property Let يمكن كذلك إنشاء إجراء Property Get

بنفس الاسم ليرجع قيمة الخاصية وتكون العبارة في أبسط الحالات هي:

```
Public Property Get propertyname As datatype
    [statements]
    propertyname = expression
End Property
```

يجب أن تكون أنواع البيانات التي يتم إرجاعها بواسطة إجراء Property Get نفس أنــواع بيانات الوسيطة التي تم تمريرها إلى إجراء Property Get بكلمات أخرى، في حالــة إعـداد خاصية لقيمة لها نوع بيانات محدد يمكن فقط قراءة قيمــة لـها نفس نـوع البيانــات يعمـل إجراءProperty Get كلما حاولت قراءة قيمة الخاصية في إجراء VBA.

ما يلي يوضح إجراء Property Get لخاصية LockedForm ويرجع ذلك الإجراء القيمــة True أو False مع نوع بيانات Boolean.

```
Public Property Get LockedForm() As Boolean
    LockedForm = Me.CompanyName.Locked
End Property
```

يختبر الإجراء خاصية Locked لعنصر تحكم البيانات على النمــوذج. إذا كـانت خاصيــة Locked لعنصر تحكم البيانات هي True يكون النموذج في وضع الاستعراض ويرجع الإجـراء القيمة True. إذا كان عنصر التحكم غير مؤمن، يرجع الإجراء القيمة False. لإرجاع قيمة مـن إجراء Property Get يتم إعداد اسم الإجراء على القيمة المراد إرجاعها كما يحدث في الدالة.

إدراج إجراء Property Get LockedForm في النموذج Suppliers ثـم احفـظ الوحـدة النمطية وقم بالتبديل إلى طريقة عرض النموذج. اكتــب؟ Forms! Suppliers.LockedForm في إطار Immediate اضغط Enter. يكتب إطار Immediate إما True أو False بناء على الوضع الذي تركت النموذج فيه.

ملحوظــة عند إنشاء خاصية مخصصــة لنمــوذج أو تقريــر باســتخدام إجــراءات Property Let وproperty Get يتم إنشاء الإجراءات كإجراءات عامــة في الوحدة النمطية للنموذج أو التقرير. إذا أردت نموذج أو تقرير أخــر أن يكون له نفس الخاصية. يجب نسخ إجراءات الخاصية للوحدة النمطية التــي تخص النموذج أو التقرير الأخر. في مثال LockedForm يمكـــن إعــادة استخدام إجراء Property Let LockedForm كما هو ويمكن نسخه إلـــى وحدة نمطية خاصة بنموذج أخر ويشـير property Get LockedForm لعنصر تحكم محدد على نموذج Suppliers ولا يمكن إعادة استخدامه دون تعديل.

استخدام إجراءات الخاصية بعد إنشاء إجراءات الخاصية مخصصة في الوحدة النمطية للنمــوذج يمكن استخدام الخاصية المخصصة بنفس أسلوب استخدام خاصية نموذج مضمنة باستثناء عــدم إمكانية إعداد أو رؤية قيمتها في ورقة خصائص. على سبيل المثال، في أثناء إدخال بيانات منتج يمكن تقديم استعراض للممولين الذين يتطابقون مع إجراءات الحدث الموضحة فيما يلي:

```
Private Sub cmdSupplier_Click
   Dim strWhere As String
   strWhere = "SupplierID = Screen.ActiveForm.SupplierID"
   DoCmd.OpenForm formname:="Suppliers",
wherecondition:=strWhere
   Screen.ActiveForm.LockedForm = True
End Sub
```

يستخدم إجراء الحدث طريقــة OpenForm لكــائن DoCmd لفتــح نمــوذج Suppliers وليتزامن مع نموذج Products. يكون النموذج المفتوح هو النمـــوذج النشــط ولــذا تســتخدم الإجراءات كائن Screen للإشارة إلى نموذج Suppliers عند إعداد خاصيــــة LockedForm على True.

لاختبار إجراء الحدث هذا، افتح Products في طريقة عرض التصميم وضــــع زر يســـمى CmdSupplier على النموذج. ادخل إجراء CmdSupplier_Click لحـــدث زر OnClick ثـــم احفظ النموذج وقم بالتبديل إلى طريقة عرض النموذج. انقر الـــزر يفتــح نمــوذج Suppliers متزامناً وفي نفس الوقت مع عناصر تحكم البيانات المؤمنة.

إنشاء طرق مخصصة لنموذج أو تقرير

يمكن إنشاء طريقة مخصصة لنموذج أو تقرير بإدخال إجراء عام في وحدة نمطيـــة لنمــوذج أو تقرير. على سبيل المثال، ينشئ Lock Controls تقوم بتأمين عناصر تحكم البيانات الموجـــودة على النموذج.

```
Public Sub LockControls()
   Dim ctl As Control
   If Me.Dirty = True Then
      RunCommand acCmdSaveRecord
   End If
   For Each ctl In Me
      If TypeOf ctl Is TextBox Then ctl.Locked = True
   Next
End Sub
```

ينشئ إجراء UnLock Controls طريقة مخصصة تسمى UnLock Contols تقوم بإلغاء تأمين عناصر تحكم البيانات.

```
Public Sub UnlockControls()
   Dim ctl As Control
   For Each ctl In Me
      If TypeOf ctl Is TextBox Then ctl.Locked = False
   Next
End Sub
```

بعد إنشاء طريقة مخصصة لنموذج بإدخال إجراء عام في الوحدة النمطية للنموذج، تستخدم الطريقة المخصصة بنفس أسلوب استخدام الطريقة المضمنة. كمثال على ذلك. أدرج إجراءات LockControls و UnlockControls الموضحة فيما سبق في الوحدة النمطية لنموذج Suppliers. احفظ الوحدة النمطية والنموذج وقم بالتبديل إلى طريقة عرض النموذج. ثم قم بتشغيل الطريقة لتأمين عناصر التحكم بإدخال Forms! Suppliers Lock Controls في إطار Immediate والضغط على Enter بذلك تكون عناصر التحكم مؤمنة. اكتب Forms! Suppliers Unlock Controls ثم اضغط Enter تصبح عناصر التحكم غير مؤمنة.

ملحوظة يمكن إنشاء طريقة مخصصة لكائن معرف بواسطة وحدة نمطية ذات فئة مستقلة بإدخال إجراء عام في الوحدة النمطية التي تقوم بالاستدعاء.

عرض أمثلة متعددة من النموذج

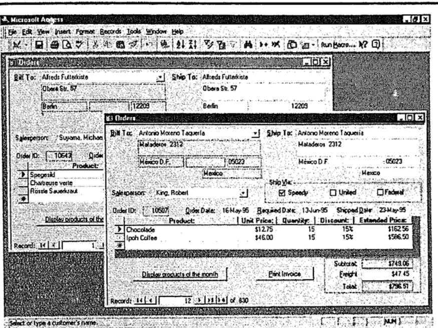
عند إنشاء نموذج جديد عن طريق وضع عناصر تحكم وإعداد خصائص في طريقة عرض تصميم النموذج وبإنشاء إجراءات في الوحدة النمطية للنموذج، يمكن إنشاء تخطيط أو مجموعة من الإرشادات لإنشاء كائن نموذج محدد. عند حفظ التخطيط يدرج اسمه في لوحة Forms في إطار Database وفي مصطلحات البرمجة التي تتعامل مع الكائنات يسمى هذا المخطط فئة (Class). عند تحديد الاسم في إطار Database ثم تنقر عليه نقراً مزدوجاً أو عند استخدام طريقة OpenForm يستخدم Access هذا المخطط لإنشاء وعرض كائن Form. في مصطلحات البرمجة التي تتعامل مع الكائنات يكون كائن Form الذي يعرض هو المثال الافتراضي default instance لهذه الفئة.

يقدم Access VBA طريقة لإنشاء أكثر من كائن Form واحد للمخطط كأمثلة إضافية للفئة. هنا يعني أنه يمكن أخذ طلبات عميلين بعرض نسختين من نموذج Orders، راجع الشكل (١٤-١١)، أو إذا كنت تراجع المنتجات في نموذج Products List يمكن عرض نماذج معلومات

المنتجات لعدة منتجات في نفس الوقت، راجع شكل (١٤-١٢). يعتبر كل من كائنات Form التي تعتمد على المخطط مثال منفصل للفئة. تقدم Access VBA كائن خاص يسمى كائن Collection كأسلوب لتتبع الأمثلة المنفصلة من الفئة.

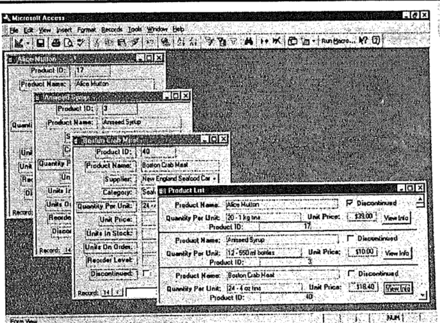
الشكل ١٤-١١

يمكن إنشاء أمثلة منفصلة من Orders والتعامل مع طلبين في نفس الوقت.



الشكل ١٤-١٢

يمكن لإنشاء أمثلة منفصلة من Products وعرض معلومات المنتجات متعددة فوراً.



الإشارة إلى نموذج مغلق في VBA

تشير إلى أي نموذج مفتوح في VBA باستخدام أي مراجع معتادة مثل: !Forms Formname, Forms ("Formname"), or Forms(n). للإشارة إلى نموذج مغلق، استخدم العبارة Form_form-name. على سبيل المثال، في إطار Immediate، يمكن إعداد قيمة خاصية Caption لنموذج Customers المغلق باستخدام عبارة التعيين التالية:

Form_Customers.Caption = "My Customers"

مثال آخر، يمكن تحديد مصدر السجل للنموذج Customers المغلق بعبارة:

?Form_Customers.RecordSource

عند الإشارة إلى نموذج مغلق باستخدام عبارة Form-formname. يفتح VBA النموذج ثم يخفيه ويعتبر المرجع إلى نموذج مغلق أيضاً مرجع إلى الوحدة النمطية للنموذج.

استخدام كائن Collection

يتيح VBA إنشاء مجموعاتك الخاصة من الكائنات والمتغيرات باستخدام كائن Collection المضمن ليحتوي على مجموعة من الكائنات من أي نوع. نعرف كائن Collection جديد ليحتوي على كائنات Form الجديدة التي نقوم بإنشائها يستخدم Access VBA الكلمة الأساسية New للإشارة إلى أمثلة جديدة بأسلوبين مختلفين: في عبارات التعريف وفي عبارات تعيين متغير الكائن.

استخدام الكلمة الأساسية New في عبارة التعريف

يمكن استخدام الكلمة الأساسية New في عبارة التعريف لمتغير كائن كما يلي:

Dim varname As New type

تعتبر Varname هي متغير الكائن الذي يتم تعريفه وtype هي نوع الكائن بدون وجود الكلمة الأساسية New تقوم عبارة التعريف بالإعلان عن متغير كائن من نوع الكائن المحدد ولكنه لا يقوم بإنشاء الكائن، يجب استخدام عبارة Set للإشارة إلى متغير الكائن الخاص بالكائن وإنشاء الكائن إن لم يكن موجوداً.

على سبيل المثال، يمكن تضمين الكلمة الأساسية New في عبارة التعريف:

```
Dim CollectForms As New Collection
```

يعين VBA بصورة آلية كائن Collection جديد أول مرة يستدعى فيها إجراء خاصية أو طريقة لمتغير كائن CollectForm. هذا يعني ذلك أنك لا تحتاج استخدام عبارة Set لتعيين كائن جديد لمتغير الكائن. عند تضمين الكلمة الأساسية New في عبارة تعريف كائن فإنك تنشئ الكائن بصورة ضمنية، على سبيل المثال، عبارة التعريف:

```
Dim tdf As New TableDef
```

تنشئ بصورة ضمنية tdf كمتغير كائن وينشئ بصورة آلية كائن TableDef في العبارة الأولى في الإجراء الذي يعين خاصية أو يشغل طريقة للكائن.

استخدام عبارة Set بدون الكلمة الأساسية New

أبسط عبارة Set يكون بناء جملتها:

```
Set objvariable = objectexpression
```

يعتبر Objvariable هو متغير الكائن الذي تعينه و Objectexpression هو مرجع لإحدى الأشياء التالية:

- متغير كائن أخر لنفس نوع الكائن.
- دالة، خاصية أو طريقة ترجع كائن من نفس نوع الكائن.

ما يلي أمثلة لعبارات Set بسيطة:

```
Dim frm As Form, rst As Recordset
Set frm = Forms!Customers
Set rst = Forms!Customers.RecordsetClone
```

تعين عبارة Set بسيطة كائن لمتغير كائن وتنشئ الكائن إن لم يكن موجوداً بالفعل. يمكن الإشارة لعدة متغيرات كائن انفس الكائن باستخدام عبارات Set.

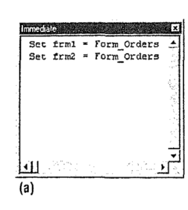
كمثال عن استخدام عبارة Set بسيطة، مع وجود نموذج Orders مغلق، اتبع الخطوات التالية:

١- ادخل Set frm1= Form_Orders في إطار Immediate واضغط Enter هذا الإجراء ينشئ متغير كائن frm1 ويجعله يشير إلى نموذج Orders، يفتح VBA النموذج ويخبئه، تأكد من ذلك باختبار Window ⇦ Unhide ثم اترك النموذج غير مخفي.

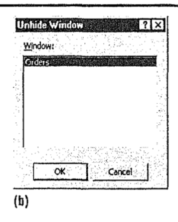
٢- ادخل Set frm2= Form_Orders في إطار Immediate واضغط Enter راجع الشكل "١٤-١٣ أ". تشير هذه العبارة متغير كائن frm2 إلى النموذج المخبأ أما

متغيرات الكائن frm1 وfrm2 فتشير إلى نفس الكائن. أغلق Window ⇦ Unhide، راجع الشكل "١٤-١٣ ب"، انقر OK لإخفاء المثال الوحيد لنموذج Orders.

٣- أغلق نموذج Orders.



(a)



(b)

الشكل ١٤-١٣ بدون الكلمة الأساسية New كل عبارة تعيين إطار Immediate "أ" تعيين متغير كائن لنفس الكائن "ب".

استخدام عبارة Set مع الكلمة الأساسية New

يمكن استخدام الكلمة الأساسية New في عبارة Set لإنشاء أمثلة جديدة وبناء الجملة هو:

Set objvariable = New objectexpression

يعتبر Objectexpression هو نفسه في عبارة Set بدون الكلمة الأساسية New.

عند تضمين الكلمة الأساسية New، تنشئ عبارة Set مثال منفصل للكائن. على سبيل المثال، يمكن استخدام العبارات الآتية لإنشاء كائنات Form منفصلة:

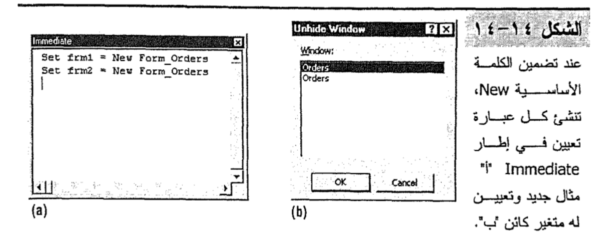
Set frm1 = New Form_Customers
Set frm2 = New Form_Customers

عند تضمين الكلمة الأساسية New، تشير متغيرات الكائن frm1 وfrm2 إلى كائنات مختلفة.

كمثال على استخدام عبارات Set البسيطة مع الكلمة الأساسية New اتبع الخطوات التالية:

١- ادخل Set frm1= New Form_Orders في إطار Immediate واضغط Enter. فتح VBA وأخفي مثال من النموذج، تأكد من صحة ذلك باختيار Window ⇦ Unhide وانرك النموذج مخفي.

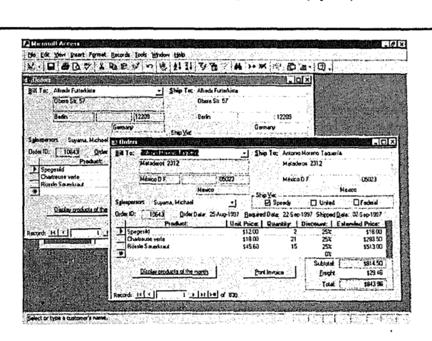
٢- ادخل Set frm2= New Form_Orders في إطار Immediate واضغط Enter، راجع الشكل "١٤-١٤ أ"، يفتح ويخفي VBA مثال ثاني من النموذج. للتأكد من صحة ذلك، اختر Window ⇦ Unhide، راجع الشكل "١٤-١٤ ب"، تدرج Orders لكل مثال منفصل. انقر OK لإلغاء إخفاء كل مثال من نموذج Orders. يوضح الشكل "١٤-١٥" المثالين.



الشكل ١٤-١٤ عند تضمين الكلمة الأساسية New، تنشئ كل عبارة تعيين في إطار Immediate "أ" مثال جديد وتعيين له متغير كائن "ب".

٣- باستخدام أزرار التنقل استعرض مجموعة سجلات كل مثال. لاحظ أن كل مثال له سجل الإشارة الحالي. عند إنشاء أمثلة متعددة لنموذج يكون لكل مثال سجل الإشارة الحالي وقد يكون له خصائص محددة باستثناء اسمه. لكل مثال لمخطط نموذج نفس الاسم وهذا يعني أنه لا يمكن الإشارة إلى مثال محدد باستخدام اسمه. عند إنشاء مثال جديد يضيف أكسس عضو لمجموعة Forms لتستطيع الإشارة للمثال بواسطة رقمه في الفهرس في مجموعة Forms.

٤- أغلق كل النماذج ما عدا مثالين نموذج Orders. ادخل Forms(0).Caption ="First" في إطار Immediate واضغط Enter. يتغير عنوان أحد الأمثلة إلى First ادخل Form(1).Caption ="Second" أغلق كلا المثالين.



الشكل ١٤-١٥ مثالين لنموذج Orders.

إنشاء أمثلة متعددة

نستكشف في هذا الجزء عملية عرض أمثلة متعددة لنموذج. أبدأ بفتح وحدة نمطية قياسية جديدة تسمى basMultipleInstances.

أنشئ متغير كائن لكائن Collection جديد بإدخال العبارة التالية في جزء Declarations في الوحدة النمطية.

```
Dim CollectForms As New Collection
```

تنشئ تلك العبارة بصورة ضمنية كائن Collection جديد يحتوي على كائنات Form. بإدخال عبارة التعريف في جزء Declaration في الوحدة النمطية فإنك تعلن عن متغير مستوى الوحدة النمطية الذي يستمر في الوجود طالما كانت قاعدة البيانات مفتوحة. فيما بعد سنرى أهمية تعريف CollectForms على مستوى الوحدة النمطية.

أدرج إجراء Multiples الموضح فيما يلي:

```
Public Sub Multiples()
    Dim frm As Form
    Set frm = New Form_Orders
    frm.Visible = True
    CollectForms.Add frm
End Sub
```

يعرف هذا الإجراء frm كمتغير كائن. تستخدم عبارة التعيين الكلمة الأساسية New لإنشاء مثال مخفي جديد لنموذج Orders. يلغي الإجراء إخفاء المثال ثم يستخدم طريقة Add لإضافة المثال لمجموعة CollectForms.

احفظ الوحدة النمطية، قم بتشغيل إجراء Multiples في إطار Immediate وبذلك يكون مثال من نموذج Orders قد تم إنشائه وعرضه وكذلك إضافته لمجموعة CollectForms.

ضع الفاصلة العليا (') أمام عبارة CollectForms. Add frm في إجراء Multiples لجعل العبارة تبدو أنها تعليق. قم بتشغيل الإجراء في إطار Immediate، ينشئ الإجراء المثال ويعرضه للحظة قصيرة. ولكن هذه المرة عندما ينتهي الإجراء ينتهي وجود متغير كائن frm. مع عدم وجود متغير يشير إليه ينتهي المثال. بإضافة المثال إلى مجموعة CollectForms. يوجد مرجع للمثال في المجموعة ليستمر وجود المثال طالما وجدت المجموعة. بالإعلان عن متغير كائن CollectForms كمتغير على مستوى الوحدة النمطية نضمن أن المجموعة وبالتالي الأمثلة الجديدة لنموذج Orders تستمر بعد انتهاء إجراء Multiples.

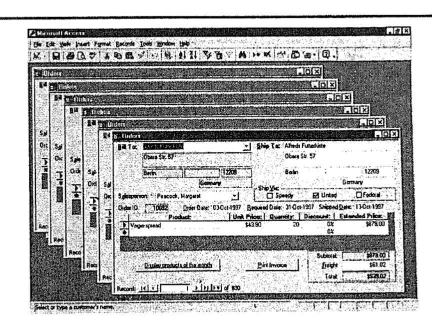
احذف الفاصلة العليا، احفظ وأغلق الوحدة النمطية. قم بتشغيل إجراء Multiples في إطار Immediate عدة مرات. كل مرة يعمل فيها الإجراء يتم إنشاء مثال جديد لنموذج Orders ويتم عرضه وإضافته إلى المجموعة. راجع الشكل "١٤-١٦".

تضمين أمثلة متعددة في تطبيق

لتوضيح كيفية تضمين أمثلة متعددة في تطبيق، يضاف زر أمر لنموذج Product List ويعين له إجراء ينشئ ويعرض مثال جديد لنموذج يعرض معلومات المنتج. إجراء الحدث للزر موضح فيما يلي:

```
Private Sub cmdView_Click()
    Dim frm As Form
    Set frm = New Form_frmViewProduct
    CollectForms.Add frm
    With frm
        .Filter = "ProductID = " & Forms![Product List]!ProductID
        .FilterOn = True
        .Caption = ProductName
        .Visible = True
    End With
End Sub
```

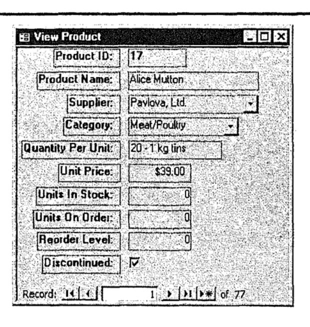
ينشئ هذا الإجراء متغير كائن frm وينشئ مثال جديد لنموذج frmViewProducts ويضيفه إلى مجموعة CollectForms. يستخدم المثال هيكل With لإعداد خاصية Filter لتزامن المثال الجديد للمنتج المحدد في نموذج Product List وأيضاً لإعداد خاصية FilterOn لتطبق على التصفية. يتضمن هيكل With أيضاً عبارات لإعداد خصائص Caption وVisible.



الشكل ١٤-١٦ إنشاء أمثلة متعددة لنموذج Orders بإضافة الأمثلة لمستوى الوحدة النمطية لكائن مجموعة CollectForms.

اتبع الخطوات الآتية لإضافة زر الأمر:

١- أنشئ نسخة من نموذج Product تسمى frmViewProduct وعدل النسخة كما هو موضح في الشكل "١٤-١٧". أغلق نموذج frmViewProduct.



الشكل ١٤-١٧ نموذج frmViewProduct.

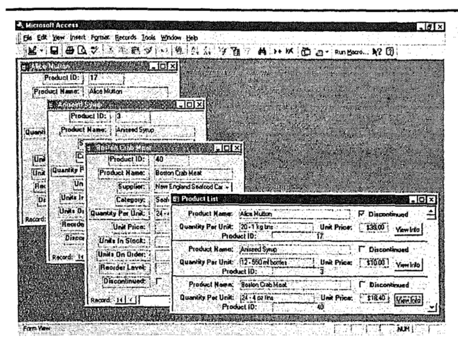
٢- افتح نموذج Product List في طريقة عرض التصميم وقم بإعداد خاصية PopUp على No. ضع زر أمر يسمى CmdView في جزء Detail وقم بإعداد خاصية Caption على View Info.

٣- انقر زر Code في شريط الأدوات. تنشأ مجموعة CollectForms ضمنياً بواسطة إدخال العبارة التالية في مقطع Declarations:

```
Dim CollectForms As New Collection
```

٤- ادخل إجراء حدث CmdView_Click الموضح فيما سبق واحفظ الوحدة النمطية.

٥- احفظ نموذج Product List وقم بالتبديل إلى طريقة عرض النموذج. حدد منتج وانقر زر View Info. يفتح مثال لنموذج frmViewProduct للمنتج الذي حددته. حدد منتج أخر وانقر زر View Info الخاص به يفتح مثال ثاني من نموذج frmViewProduct للمنتج الثاني. يوضح الشكل "١٤-١٨" نتائج عرض أمثلة متعددة لنموذج frmViewProduct.



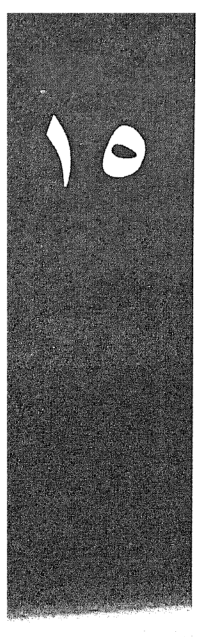
الشكل ١٤-١٨
عرض أمثلة متعددة من نموذج كل منها يتزامن مع منتج محدد.

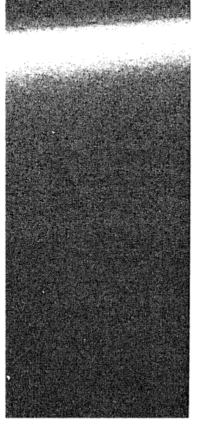
خلاصة

قدم هذا الفصل تقنيات إنشاء وتعديل كائنات جديدة بطريقة برمجية وما يلي هو أهم نقاط هذا الفصل:

- في معظم الحالات، تستخدم طريقة ...Create لكائن وصول البيانات لإنشاء كائن تابع. عادة يجب إنشاء كائن تابع كامل بالخصائص المطلوبة. وغالباً مع الكائن التابع المطلوب له قبل أن تحفظ الكائن التابع. على سبيل المثال، إنشاء كائن TableDef؟ يجب إعطاء الكائن اسم صحيح وإنشاء على الأقل كائن Field واحد له خصائص Name وType.
- إنشاء خاصية مخصصة لكائن وصول البيانات باستخدام طريقة Create Property الخاصة بالكائن وإرفاق كائن Property جديد لمجموعة Properties الخاصة بالكائن.
- ينشئ تطبيق مثل أكسس يعمل كمضيف لمحرك قاعدة البيانات Jet خصائص معرفة بواسطة التطبيق لكائن وصول البيانات. لا يكون Jet على دراية بالخصائص المعرفة بواسطة التطبيق إلى أن يتم إعداد الخاصية في واجهة التطبيق أو إنشاء كائن Property للخاصية وإرفاق كائن Property الجديد لمجموعة Properties لكائن وصول البيانات.
- يقدم أكسس VBA مجموعة من الدوال المضمنة الممكن استخدامها إنشاء نماذج وتقارير عناصر تحكم جديدة برمجياً.
- يقدم أكسس ٢٠٠٠ كائن Module بخصائص وطرق يمكن استخدامها إنشاء وتعديل الوحدات النمطية برمجياً.

- يتم إنشاء خاصية مخصصة لنموذج أو تقرير بإنشاء زوج من إجراءات Property Let أو Property Set لخاصية ترجع الكائن وإجراء Property Get في الوحدة النمطية للنموذج أو التقرير يتم إعداد الخاصية المخصصة باستخدام عبارة تعيين في الإجراء وتقرأ القيمة للخاصية المخصصة بالإشارة إلى الخاصية المخصصة في الإجراء.
- يتم إنشاء طريقة مخصصة لنموذج أو تقرير بإنشاء دالة عامة أو إجراء فرعي في الوحدة النمطية للنموذج أو التقرير.
- يقدم أكسس ٢٠٠٠ وحدة نمطية ذات فئة مستقلة يمكن استخدامها لإنشاء كائنات جديدة غير مرتبطة بالنماذج والتقارير. أنشئ خصائص مخصصة للفئة بإنشاء إجراءات خصائص Property Let وProperty Set وProperty Get وأنشئ طرق بإنشاء دوال عامة أو إجراءات فرعية وتخزين الإجراءات في الوحدة النمطية للفئة.
- عند إنشاء نموذج أو تقرير جديد، فإنك تنشئ مخطط، أو فئة Class، يستخدمها أكسس لإنشاء كائنات Report أو Form منفردة. عند فتح نموذج ينشئ أكسس المثال الافتراضي. يمكن استخدام المخطط لإنشاء أمثلة منفصلة باستخدام الكلمة الأساسية New في عبارات التعيين. يمكن إنشاء كائن Collection على مستوى الوحدة النمطية لتتبع الأمثلة المتعددة.





توسيع أكسس

يعتبر هذا الفصل الأخير مراجعة عامة على الموضوعات المتقدمة المتاحة مع برمجة VBA. ويبدأ حيث يبدأ بمناقشة كيفية تحويل الماكرو إلى إجراءات VBA مما يتيح لك الاستفادة منها.

كما يقدم هذا الفصل بعض الطرق التي توضح المكتبات المخصصة التي تتيح لك إعادة استخدام إجراءات VBA في قواعد البيانات الأخرى. كما أنك سوف تتعلم كيفية إنشاء قواعد البيانات المكتبية هذه وكيفية لوصول إليها من قواعد بيانات أخرى. كما أنك سوف تتعلم المزيد حول مكتبات الارتباطات الديناميكية "DLL" التي تخزن التعليمات البرمجية مع تصحيحها بالتفصيل من قبل مبرمجين آخرين. ومن أحد أهم مكتبات الارتباطات الديناميكية هي التي تتيحها ويندوز المسماه Windows API. ويمكنك زيادة قدرات التطبيقات عن طريق استخدام استعارة واستخدام التعليمات البرمجية الخاصة بهذه المكتبات.

ثم يوضح الفصل كيفية استخدام ActiveX لتعزيز قاعدة البيانات وهو عبارة عن مجموعة من التقنيات التي تسمح بالاتصال بين أجهزة الحاسبات والتطبيقات والكائنات. وسوف تتعلم أيضاً كيفية إضافة عناصر تحكم جديدة إلى أكسس. وهي تسلك نفس سلوك عناصر تحكم مربع الأدوات العادي. إلا أنها تقدم الإمكانات التي لا يقدمها أكسس. كما أن هذا الفصل يعرض أيضاً كيفية كتابة برامج VBA التي تتحكم في الكائنات الخاصة بالتطبيقات الأخرى مثل إكسل أو وورد. وتسمى التقنية التي تتيح لك التحكم في أحد التطبيقات الأخرى بشكل برمجي هي Automation.

ملاحظة تستخدم الكلمة تطبيق للإشارة إلى البرامج الشائعة مثل وورد إكسل وما إلى ذلك. أما الكلمة "مشروع" فإنها تشير إلى مجموعة من إجراءات فيجوال بيسك التي تقوم بكتابتها لملف معين الذي تم إنشاؤه باستخدام أحد هذه البرامج.

تحويل الماكرو إلى إجراءات VBA

تعلمت في هذا الكتاب كيفية استخدام برمجة VBA لجعل العديد من مهام قاعدة البيانات تتم بشكل آلي. وعلى جانب آخر فإنه يمكنك بناء مشروعات معقدة ومتميزة باستخدام الماكرو فقط بدلاً من VBA. إلا أن المشروعات قد تحتاج إلى إجراءات VBA حتى ولو كانت في غاية الصغر. إذا ما أرد تضمين أياً من مميزات VBA التي تمت الإشارة إليها في هذا الكتاب، وتتحكم في ذلك نوعية المهمة التي تريد جعلها آلية والشخص الذي سوف يستخدمه.

يمكن إعادة كتابة بعض أو كل الماكرو على هيئة إجراءات VBA. كما أن أكسس يقدم إمكانية تحويل الماكرو إلى تعليمات VBA البرمجية مع إضافة بعض إمكانات معالجة الأخطاء. وبعد القيام بهذه العملية يمكنك تعديل تعليمات معالجة الأخطاء. غير بنيات القرارات والحلقات إلى المميزات الموجودة في VBA والتي تتميز بأن لها قدرات أكثر.

وهناك طريقتان لعمل مثل هذه العملية:

- يمكن تحويل كل الماكرو الخاصة بالحدث لأحد النماذج أو التقارير لمجموعة من الإجراءات التي يقوم أكسس بحفظها في وحدات النموذج أو التقرير.
- يمكن تحويل كل الماكرو في ورقة الماكرو إلى إجراءات وظيفية مخزنة في الوحدة الجديدة القياسية.

وقبل الدخول إلى المزيد من التفاصيل فيما يخص هذا الموضوع يجب أولاً معرفة الأسباب التي تجعلك تحتاج إلى تحويل الماكرو إلى إجراءات VBA.

ملاحظة للمزيد من المعلومات حول جعل قاعدة البيانات آلية باستخدام الماكرو اقرأ *Mastering Access 2000 Premium Edition*.

أسباب تحويل الماكرو

فيما يلي بعضاً من هذه الأسباب:

لعمل الوظائف المخصصة: يتضمن أكسس مجموعة ضخمة من الوظائف المضمنة التي يمكن استخدامها مع المعاملات لإنشاء تعبيرات في غاية التعقيد وأحياناً يكون التعبير الذي تريد استخدامه طويلاً ومعقداً. وبدلاً من إعادة كتابته يمكنك إنشاء الوظائف المخصصة الخاصة بك. كما يمكن استخدام الوظائف المخصصة بنفس الطريقة التي تستخدم بها الوظائف المضمنة.

للتعامل مع القرارات المعقدة: تقدم وظائف القرار المضمنة وشروط الماكرو وإجراء RunMacro للعمليات المكررة طرقاً ملائمة للتعامل مع القرارات البسيطة. إذا كنت تحتاج إلى بنيات قرار أكثر تعقيداً فإنه يمكنك استخدام الماكرو، إلا أن استخدام بنيات عناصر التحكم الخاصة برنامج VBA دائماً ما تكون أفضل وأسرع.

لتصيد الأخطاء: حيث أنها تعوض نقص إمكانات معالجة الأخطاء في برمجة الماكرو، بما تتميز به من إمكانية التعامل مع الأخطاء التي لا يمكن التنبؤ بها. ويمكن تضمين معالجات الأخطاء بطريقتين: بطريقة فردية على الإجراء حيث يتعامل مع الخطأ عند تشغيل الإجراء أو في إجراء الحدث بالنسبة لحدث Error الخاص بالنماذج والتقارير للتعامل مع أخطاء محرك قاعدة البيانات أو واجهة المستخدم التي تحدث عند نشاط التقرير أو النموذج.

لعمل نسخة مطبوعة: لا يمكن استخدام الوثائق الخاصة بالماكرو والتي قام Database Documenter بنفس السهولة التي يمكن بها استخدام تعليمات VBA البرمجية.

لإنشاء الكائنات والتعامل معها: يمكنك في أغلب الأحوال إنشاء الكائنات والتعامل معها. إلا أنه في بعض الأحيان تكون هناك حاجة إلى تحديد أو تعيين أحد الكائنات بشكل آلي عند تشغيل التطبيق. وربما تحتاج إلى إنشاء معالجات مخصصة تكون خاصة بالتطبيق. يمكن استخدام استعلام تعريف البيانات الخاص بأكسس SQL لإنشاء وتعديل الجداول أو عمل الفهارس أو إلغائها أو عمل علاقات بين الجداول بعضها وبعض. ويمكن مع VBA إنشاء أي كائن في قاعدة البيانات والتعامل معه بالإضافة إلى قاعدة البيانات نفسها.

لعمل الإجراءات التي لا تكون متاحة باستخدام الماكرو: هناك بعض الإجراءات التي لا يمكن تنفيذها بالماكرو فقط. مثل معالجة المعاملات وهي متاحة في VBA فقط. حيث أن كل العمليات تتم في الذاكرة وإذا فشلت هذه العمليات فإنه يتم تجنب العملية مع ترك الجداول كما كانت قبل بدء المعاملة. أما إذا تمت كل المعاملات على نحو ناجح يتم تحديث جداول البيانات بالنتائج.

ملاحظة تحتاج قواعد البيانات المبنية على VBA استخدام الماكرو لعمليتين: ماكرو AutoKeys لإعادة تعيين المفاتيح وماكرو AutoExec لتنفيذ بعض الإجراءات عند بداية تشغيل قاعدة البيانات.

لتمرير الوسائط إلى التعليمات البرمجية: يتم تعيين وسائط الإجراء عند إنشاء الماكرو. ولا يمكن تغييرها في أثناء التشغيل، كما أن من غير الممكن استخدام المتغيرات. وفي بعض الأحيان يمكن استخدام بعض المراجع إلى كائن Screen والتي لها نفس اثر استخدام المتغير. إلا أن VBA يقدم طرقاً إضافية أكثر فائدة لتغيير الوسائط الموجودة في التعليمات البرمجية الخاصة مع استخدام المتغيرات في الوظائف.

لإنشاء المتغيرات: الطريقة الوحيدة لإنشاء المتغير في الماكرو هي استخدام عنصر تحكم غير منضم على النموذج للحفاظ على القيمة المؤقتة. وبينما يكون النموذج مفتوحاً تكون القيمة الموجودة في عنصر التحكم متاحة للاستخدام عن طريق النماذج والاستعلامات وإجراءات VBA الأخرى. ويوفر VBA القدرة على إنشاء المتغير في الذاكرة بدون استخدام عنصر تحكم من النموذج. كما يتيح لك VBA أيضاً تقليل مجال المتغير عن طريق تحديد أي الإجراءات الأخرى التي يمكنها الوصول إلى المتغير.

لتحسين الأداء: هناك بعض المميزات في VBA يمكن من خلالها تحسين الأداء مثل طريقة Seek. كما أنه يقدم الإشارات المرجعية لحفظ المكان في مجموعة السجلات مما يمكنك من العودة سريعاً إلى السجل،كما أن معالجة المعاملات تتم بطرقة أسرع. كما أن إجراءات VBA تتميز بالسرعة لأنه قد تم تجميعها أما الماكرو الذي تقوم بإنشائه فغير مجمع.

للعمل خارج أكسس: يمكن مع الماكرو استخدام RunApp لبدء برنامج من الدوس والويندوز. إلا أنه بمجرد بدء التطبيق فإنه يمكن التعامل معها على نحو مكثف. ويمكن جعل التطبيقات الأخرى تتم بطريقة آلية باستخدام Automation أو Dynamic Data Exchange التي يرمز لها بالرمز "DDE". كما يمكن كتابة التعليمات لعناصر تحكم ActiveX كما أنه باستخدام VBA يمكن مد نطاق البرمجة إلى أكثر من أكسس. ويمكن جعل التطبيقات الأخرى تتعامل بطريقة آلية باستخدام Automation أو Dynamic Data Exchange التي يرمز لها بالرمز "DDE" كما يمكن كتابة التعليمات الخاصة بعناصر تحكم ActiveX. كما يمكن التعامل مع قواعد البيانات الخارجية مثل ODBC أو OLE DB. كما يمكن أيضاً الاتصال مع API باستخدام الوظائف الموجودة في DLL.

ويمكن استخدام أياً من الماكرو أو VBA لجعل قاعدة البيانات تعمل بطريقة آلية، على الرغم من أن أغلب المبرمجين يستخدمون VBA كما أن البعض الآخر يستخدم كل منهما.



قم بإنشاء نسخة جديدة من Northwind وسمها Northwind_Ch15.

تبديل ماكرو الحدث إلى ماكرو الإجراء

يمكن تبديل كل ماكرو الحدث الخاص بالنموذج أو تقرير إلى ماكرو الإجراء مع تخزينها في وحدة التقرير أو النموذج. وعند القيام بذلك يقوم أكسس بإنشاء وحدة التقرير أو النموذج مع تغيير اسم ماكرو الحدث إلى إجراء الحدث المناظر باستخدام الصيغة التالية:

```
Form_eventname
controlname_eventname
```

أو

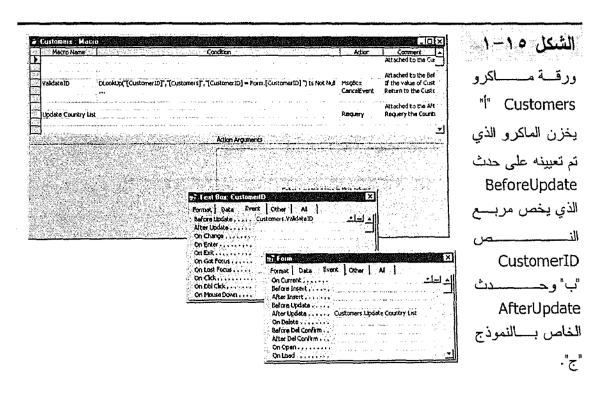
```
Report_eventname
controlname_eventname
```

يقوم أكسس بتعيين إجراء الحدث إلى خاصية الحدث. وإذا تواجدت الوحدة النمطية الخاصة بالتقرير أو النموذج فإن أكسس يضيف الإجراءات المحولة إلى الوحدة النمطية. وتحتوي ورقة الماكرو التي يتم فيها تخزين ماكرو الحدث مع عدم تغييرها.

ولمعرفة هذه العملية افتح قاعدة بيانات Northwind_Ch15 ثم تتبع الخطوات التالية:

١-افتح ورقة الماكرو Customers. يوضح الشكل ١٥-١أ ماكرو الحدثين لنموذج Customers: يتم تعيين ماكرو ValidateID على حدث BeforeUpdate الخاص بحقل CustomerID كما أن ماكرو Update Country List يتم تعيينه إلى حدث AfterUpdate الخاص بالنموذج.

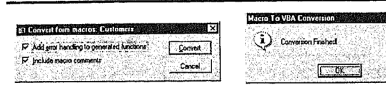
٢-افتح ورقة نمط Customers. ويوضح الشكل ١٥-١أ و١٥-١ج تعيينات الماكرو لخصائص الحدث. قم بالتمرير إلى نهاية ورقة خاصية Form لملاحظة أن خاصية HasModule قد تم تعيينها على No مما يعني أن نموذج Customers هو نموذج لا يحتوي على وحدة نمطية.



الشكل ١٥-١ ورقة ماكرو Customers "أ" يخزن الماكرو الذي تم تعيينه على حدث BeforeUpdate الذي يخص مربع النص CustomerID و"ب" حدث AfterUpdate الخاص بالنموذج "ج".

٣-اختر Tools ⇦ Macro ⇦ Visual Basic Convert Form's Macros to. يوضح الشكل "١٥-٢ أ" حوار Convert Form Macros مع الخيارات الخاصة بإضافة معالجات الأخطاء البسيطة مع تضمين تعليقات الماكرو كتعليقات في إجراء VBA.

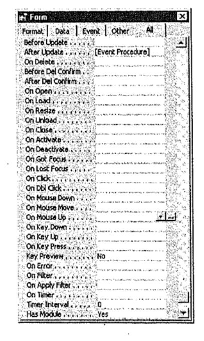
٤-انقر زر Convert مع اختيار كل من الخيارين وبعد بعض الثواني تظهر الرسالة الموضحة في الشكل "١٥-٢ ب". وتشير أوراق الخاصية لكل من النموذج ومربع النص Customer ID إلى أن تعيينات الماكرو تم تبديلها بالتعيينات الخاصة بإجراءات الحدث كما أن خاصية HasModule تم تغييرها إلى Yes "انظر الشكل ١٥-٣".



الشكل ١٥-٢

استخدام حوار Convert Form Macros لإضافة معالجات الأخطاء وتعليقات الماكرو إلى إجراء الحدث الجديد "أ". يعرض أكسس رسالة تتيح لك معرفة أن التحويل تم بشكل صحيح.

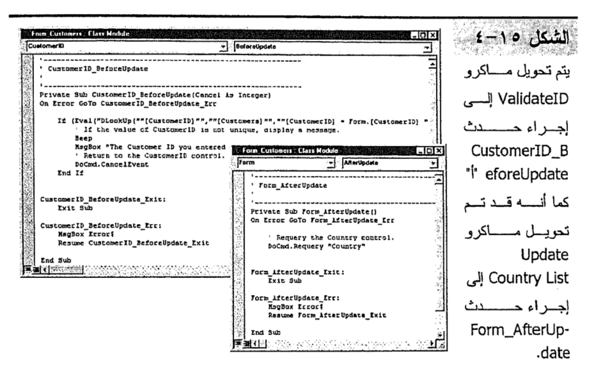




الشكل ١٥-٣

عند تحويل ماكرو النموذج فإن تعيينات الماكرو يتم تبديلها بالتعيينات الخاصة بإجراءات الحدث "أ" مع تعيين خاصية HasModule على Yes "ب".

٥- انقر زر Code في شريط الأدوات. وقد تم تحويل ماكرو الحدث ValidateID إلى إجراء الحدث CustomerID_BeforeUpdate الموضح في الشكل ١٥-٤أ كما أنه قد تم تحويل ماكرو Update Country List إلى إجراء حدث Form_AfterUpdate الموضح في الشكل ١٥-٤ب كما تتضمن إجراءات الحدث الجديد تعليقات الماكرو. ويحتوي الإجراء الجديد على معالجات الأخطاء الأساسية التي تعرض التعليمات البرمجية الخاصة بالماكرو في مربع الرسالة إذا حدث أحد الأخطاء في أثناء التشغيل.



الشكل ١٥-٤ يتم تحويل ماكرو ValidateID إلى إجراء حدث CustomerID_BeforeUpdate "أ" كما أنه قد تم تحويل ماكرو Update Country List إلى إجراء حدث Form_AfterUpdate.

تحويل الماكرو إلى إجراءات الوظيفة

بدلاً من تحويل ماكرو الحدث الخاص بالنموذج أو التقرير إلى إجراءات الحدث، يمكن تحويل كل الماكرو المخزن في ورقة الماكرو إلى إجراءات الوظيفة مخزنة في الوحدة النمطية الجديدة. وفي هذه الحالة يتم تحويل الماكرو إلى إجراء الوظيفة إلا أنه لا يتم تعيينه إلى الحدث. وبدلاً من ذلك فإنه يتم تعيين الماكرو الأصلي إلى الحدث كما أنه يجب إعادة تعيين خاصية الحدث لبدء تشغيل إجراء الوظيفة. وأيضاً فإن الماكرو الذي يتم استدعاؤه بماكرو آخر باستخدام ماكرو RunMacro فإنه يتم تحويله إلى إجراء الوظيفة. ولا يتم تعديل الماكرو أو الإجراء الذي يقوم باستدعاء الماكرو الأصلي آلياً، فيجب عمل ذلك يدوياً.

وعند تحويل الماكرو إلى إجراءات الوظيفة في الوحدة النمطية القياسية فإن VBA يذكر الماكرو الإجراء باستخدام الصيغة:

macrosheetname_macroname

حيث تقوم بوضع الرمز "_" بدلاً من المسافة في كل من الاسمين. يجب تغيير الخصائص أو العبارات التي تقوم باستدعاء إجراءات الوظيفة.

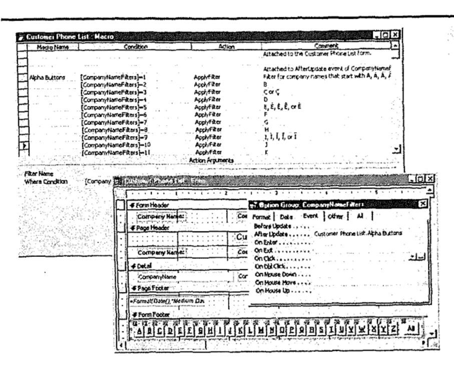
وللتعرف على هذه العملية تبع الخطوات التالية:

١- افتح ورقة الماكرو Customer Phone List. كما أن ورقة النمط تحتوي على ماكرو حدث Alpha Buttons، كما هو موضح في الشكل التالي ثم قم بالتمرير لأسفل لملاحظة ماكرو Print وهو الذي يتم استدعاؤه كماكرو مساعد عن طريق استخدام الماكرو p^ في ورقة النمط Sample Autokeys.

ملاحظة يستخدم الماكرو Alpha Buttons خاصية RecordsetClone للإشارة إلى مجموعة سجلات النموذج، ثم يستخدم خاصية RecrdCount لحساب عدد السجلات المعروضة في النموذج. وللإشارة إلى خاصية لمجموعة السجلات في الماكرو. استخدم الصيغة Recordsetclone.propertyname.

٢- اختر File ⇦ Save As ثم اختر وحدة نمطية من القائمة المنسدلة ثم اكتب Customer Phone List في حقل Name (انظر الشكل ١٥-٦أ ثم انقر Ok. انقر خيارات إضافة معالجة الأخطاء والتعليقات في Convert Macro. فيقوم أكسس بإنشاء وحدة نمطية جديدة تسمى Converted Macro - Customer Phone List، مع عرض رسالة توضح أن التحويل تم بالفعل على نحو سليم.

٣- اختر Converted Macro - Customer Phone List من علامة تبويب Modules الخاصة بإطار Database ثم انقر Design لعرض الوحدة النمطية الجديدة. يوضح الشكل ١٥-٧ إجراء الوظيفة Customer_Phone_List_Alpha_Buttons. لاحظ أن الإجراء يستخدم خاصية CodeContextObject من كائن التطبيق للإشارة إلى الكائن الخاص عند تشغيل الإجراء.



الشكل ١٥-٥

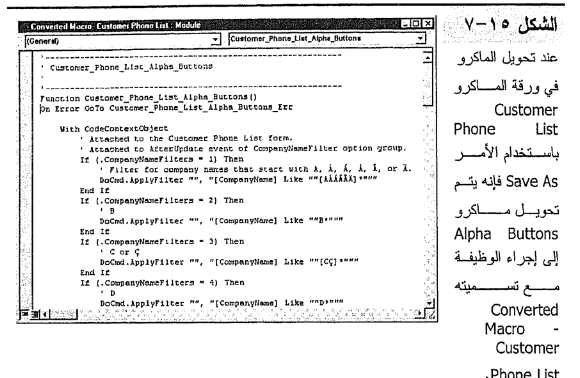
ورقة الماكرو Customer Phone List "أ" تحتوي على ماكرو حدث Alpha Buttons المعين على حدث AfterUpdate لمجموعة الكائن في نموذج Customer Phone List "ب".

الشكل ١٥-٦

استخدم الحوار Save As لحفظ ورقة الماكرو كوحدة قياسية جديدة "أ" وحوار ماكرو Convert لإضافة معالجات الأخطاء والتعليقات "ب".

٤- افتح نموذج Customer Phone List في عرض Design. وتكون خاصية AfterUpdate الخاصة بمجموعة خيار CompanyNameFilter معينة على ماكرو الحدث. ولتعيين إجراء الوظيفة بدلاً من الماكرو اكتب =Customer_Phone_List_Alpha_Buttons() في مربع خاصية AfterUpdate.

٥- احفظ النموذج ثم انتقل إلى عرض Form. انقر أحد الأزرار في مجموعة خيار النموذج.

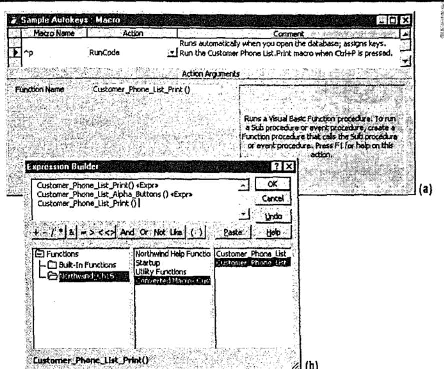


الشكل ١٥–٧ عند تحويل الماكرو في ورقة الماكرو Customer Phone List باستخدام الأمر Save As فإنه يتم تحويل ماكرو Alpha Buttons إلى إجراء الوظيفة مع تسميته Converted Macro - Customer Phone List.

٦– افتح ورقة النمط Sample Autokeys. ويستخدم الماكرو ^p إجراء RunMacro لتشغيل ماكرو Print. غير إجراء الماكرو إلى RunCode. اكتب Customer_Phone_List_Print() كوسيطة اسم الوظيفة "انظر الشكل ١٥–٨". ويجب حذف علامة يساوي إلا أنه يجب وضع الأقواس حتى إذا كانت الوظيفة لا تحتوي على أية وسائط.

أو يمكن النقر على وسيطة اسم الوظيفة ثم نقر زر Build الموجود على يمين مربع الوسيطة لعرض Expression Builder. انقر مجلد Functions ثم انقر المجلد Northwind_Ch15. يذكر Expression Builder الوحدات النمطية القياسية في مربع القائمة في المنتصف "انظر الشكل ١٥–٨ب". اختر Converted Macro - Customer Phone List من المربع الثاني وCustomer_Phone_List_Print() من المربع الثالث انقر Paste ثم انقر OK.

٧– احفظ ورقة الماكرو. انقر نموذج Customer Phone List ثم انقر أحد الأزرار من مجموعة خيار النموذج ثم اضغط على Ctrl+P. يقوم الماكرو ^p بتشغيل الإجراء.



الشكل ١٥-٨ لاستدعاء إجراء الوظيفة، غير حدث ماكرو Run Macro إلى حدث ماكرو Run Code "أ". استخدم Expression Builder لتعيين Function وسيطة Name لحدث ماكرو Run Code "ب".

فهم مكتبة قاعدة البيانات

عند إنشاء إجراء عام تريد إعادة استخدامه في مشروعات أكسس الأخرى فإنه يمكن جعلها متاحة بطريقتين: عن طريق تخزينها في أحد الوحدات النمطية ثم مع استيرادها إلى مشروع آخر، أو إنشاء مكتبة قاعدة بيانات مع جعلها متاحة للمشروع الآخر. ومكتبة قاعدة البيانات هي مجموعة من الإجراءات والكائنات التي يمكن استدعاؤها من مشروع آخر لأكسس.

إنشاء مكتبة قاعدة بيانات

ويشبه إنشاء قاعدة بيانات عادية في أكسس إلا أنك لن تقوم بتخزين البيانات في مكتبة قاعدة البيانات بنفس الطريقة التي تقوم بها عند التعامل مع قاعدة البيانات العادية. يمكن استخدام أي اسم صالح للمكتبة والتحويل هو استخدام الملحق mda. لمكتبة قاعدة البيانات، إلا أنه يمكن استخدام الملحق mdb.

اختبار الإجراءات

قبل وضع ماكرو أو إجراء VBA في مكتبة قاعدة البيانات يستحسن أن تقوم باختبارها على نحو كامل. ويجب أن يحتوي إجراء VBA على معالج للأخطاء للتعامل مع الأخطاء غير المتوقعة.

برمجة الكفاءة

أعد فحص البرنامج لأفضل أداء بعد إجراء الاختبار على الإجراء أو الماكرو. وفيما يلي بعض الاقتراحات:

- عند القدرة على اختيار بنية عنصر التحكم تأكد أنك استخدمت أفضلها. فمثلاً إذا كنت تقوم بعمل جولة في المجموعة استخدم For Each...Next بدلاً من For...Next.
- تجنب وظيفة القرار IIF وChoose وSwitch واستخدم بنيات If...Then وSelect Case.
- استخدم المتغيرات التي لها أنواع معينة من البيانات وأنواع بيانات الكائن.
- استخدم افضل الخيارات للإشارة إلى الكائنات. مثل Me للإشارة إلى النموذج الذي يتم فيه تشغيل النموذج. استخدم بنية With...End With لتعيين العديد من الخصائص.
- احتفظ بنسخة من الماكرو والإجراء مع إخراج التعليقات من النسخ الموجودة في المكتبة. ويجب أن يكون الإجراء قادراً على التعامل مع الأخطاء. وعن طريق عمل النسخة يمكن تغيير الإجراء في أي وقت.

تجميع الإجراءات

يمكن تحقيق أسرع معدل لتعليمات VBA البرمجية عند تشغيلها على نحو مجمع. يجب تجميع كل الإجراءات عند إضافتها إلى المكتبة أو بعد إضافة الإجراء الجديد أو إدخال بعض التعليمات عليه. ولتجميع أحد الإجراءات افتح أي وحدة نمطية في مكتبة قاعدة البيانات ثم اختر Debug ⇦ Compile "اسم المكتبة".

تحذير يمكن حفظ مكتبة قاعدة البيانات أو أي قاعدة بيانات أخرى في أكسس ٢٠٠٠ كملفات mde. وهو الذي يقوم بتجميع الوحدات النمطية مع حذف تعليمات البرمجة الأصلية التي يمكن قراءتها. مع ضغط قاعدة البيانات. مما يؤدي إلى تقليل حجم ملفات mde. كما أن المشروع يستخدم الذاكرة بطريقة أفضل وبأداء أفضل.

إنشاء مرجع لمكتبة قاعدة البيانات

قبل بدء استخدام مكتبة قاعدة البيانات في مشروع أكسس فإنك بحاجة إلى إضافة أحد المراجع إلى مكتبة قاعدة البيانات. وفي أول مرة يقوم المشروع باستدعاء الإجراء لأول مرة في المكتبة

فإن أكسس يقوم بتحميل الوحدات النمطية التي تحتوي على الإجراءات في الذاكرة. ويمكن استدعاء إجراءات المكتبة كما لو كانت جزءاً من المشروع.

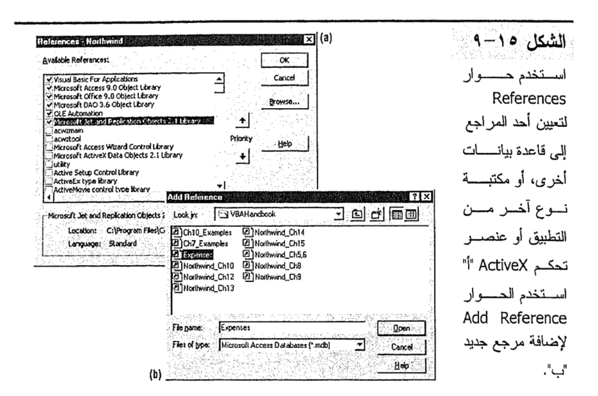
سوف نقوم بإضافة أحد المراجع إلى قاعدة البيانات Expenses.mdb لفهم كيفية حدوث ذلك لذلك يمكن استخدامها كمكتبة لقاعدة البيانات في قاعدة البيانات Northwind_Ch15.

١- افتح أي وحدة نمطية في قاعدة البيانات الحالية ثم اختر Tools ⇦ Reference لعرض الحوار Reference "انظر الشكل ١٥-٩ أ" وتكون قاعدة البيانات الحالية قادرة على الوصول إلى العناصر الحالية التي تم اختيارها. استخدم أزرار Priority لتغيير موضع أحد العناصر في القائمة. وإذا لم يتم ذكر العنصر يمكن إضافته.

٢- انقر زر Browse ثم اختر Database Access Microsoft "*.mdb" في قائمة لتحرير والسرد. لحوار Add Reference "انظر الشكل ١٥-٩ ب".

٣- ضع الملف Expenses.mdb في فهرس C:\VBAHandbook مع نقر Open. تظهر الآن قاعدة البيانات Expenses.mdb وهي مختار في جهوار Refrences. وهي تحتوي على الوحدة النمطية IsLoadedModule، التي تحتوي على وظيفة IsLoaded التي تستعيد القيمة True إذا كانت القيمة التي تقوم بتمريرها إلى الوظيفة على هيئة سلسلة كانت مفتوحة. وإلا فإنها تقوم برد القيمة إلى False. وتظهر القيمة نفسها في الوحدة النمطية Utility Function في Northwind_Ch15.

٤- اغلق حوار References. انقر إطار Database. اختر الوحدة النمطية Utility Function.



الشكل ١٥-٩

استخدم حوار References لتعيين أحد المراجع إلى قاعدة بيانات أخرى، أو مكتبة نوع آخر من التطبيق أو عنصر تحكم ActiveX "أ" استخدم الحوار Add Reference لإضافة مرجع جديد "ب".

٥- افتح نموذج Categories مع عرض الإطار الحالي اكتب ?IsLoaded ثم اضغط Enter. فيقوم أكسس بوضع وظيفة IsLoaded في قاعدة بيانات Expenses.mdb وتشغيل إجراء الوظيفة مع استعادة القيمة -١ إلى True.

تحذير يتيح لك أكسس ٢٠٠٠ إنشاء مراجع عن طريق البرمجة باستخدام كائن References. ابحث عن كائن المرجع للمزيد من المعلومات.

إذا قمت بتغيير مكان مكتبة قاعدة البيانات أو تغيير اسمها فإن أكسس قد لا يستطيع العثور عليها. وعند استدعاء أحد الإجراءات في مكتبة قاعدة البيانات ثم يقوم بعرض رسالة الخطأ ثم يقوم بفتح الحوار Reference التي تعرض الكلمة "MISSING" في مقدمة اسم مكتبة قاعدة البيانات. يمكن الاستعراض للعثور على مكان جديد مع إعادة تعيين المرجع.

وإذا قمت بإلغاء اختيار أحد العناصر في حوار References فإن قاعدة البيانات لا تستطيع التعرف على العنصر. ولتجنب هذه الأخطاء يجب حذف كل المراجع في قاعدة البيانات لقاعدة بيانات العنصر أو الإجراء.

تحرير التعليمات البرمجية الخاصة بالمكتبة

يمكن تحرير إجراء VBA المخزن في مكتبة قاعدة البيانات في أثناء العمل في المشروع الحالي بدون فتح مكتبة قاعدة البيانات على نحو مباشر.

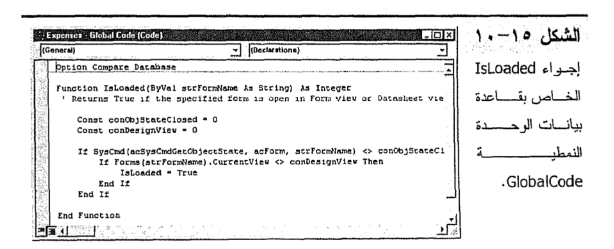
لعرض الإجراء في مكتبة قاعدة البيانات افتح الوحدة النمطية من Project Browser فمثلاً لعرض إجراء وظيفة IsLoaded تتبع الخطوات التالية:

١- افتح Project Browser ثم قم بتوسيع قاعدة بيانات Expenses من قائمة السرد والتحرير Project/Library.

٢- قم بتوسيع فهرس Modules في Project Browser.

٣- انقر أ نقراً مزدوجاً الوحدة النمطية Global Code.

٤- افتح الإجراء IsLoaded الخاص بالوحدة النمطية عن طريق اختيارها من مربع السرد والتحرير الموجود على الجانب الأيمن من إطار Module "انظر الشكل ١٥-١٠".
والآن يمكنك عمل بعض التغييرات الخاصة بالإجراء.



الشكل ١٥-١٠ إجراء IsLoaded الخاص بقاعدة بيانات الوحدة النمطية GlobalCode.

فهم مكتبات الارتباط الديناميكي

وهي عبارة عن مكتبة تحتوي على العديد من الإجراءات التي يمكن أن يرتبط بها أكسس وإكسل. وهي توجد في الملفات على نحو منفصل عن التطبيق. ويمكن أن يتم ربط العديد من التطبيقات بمكتبة واحدة في نفس الوقت. وتسمى الرابطة ديناميكية لأن الرابطة تكون نشطة في وقت التشغيل فقط عندما يتم استدعاء الإجراء الموجود في المكتبة. دائماً ما تكتب DLL في لغة C أو C++ للوصول إلى الأداء المثالي. كما أنها دائماً ما يكون لها الملحق dll.. يقوم نظام التشغيل ويندوز بتحميل DLL عند الحاجة إليها.

هناك نوعان من DLL: المكتبات التي يتم تحديدها بمكتبة النوع وأخرى ليس لها مكتبة نوع. والفرق بين هذين النوعين يتضح عند كتابة تعليمات VBA البرمجية الخاصة بكل منهما.

استخدام مكتبة النوع

وهي عبارة عن ملف يتم اتصاله مع DLL إلا أنه يكون منفصلاً عنه. وهو يحتوي على المعلومات الخاصة بالإجراءات في DLL. يستخدم VBA أحد تقنيات ActiveX التي تسمى Automation للتعامل مع مكتبة النوع كما أن مكتبة النوع دائماً ما يكون لها الملحق olb. أو tlb.. وعندما يحتوي DLL على مكتبة نوع مرتبطة به فإن برنامج التثبيت يقوم بتثبيت مكتبة النوع في جزء OLE من Windows Registry.

وإذا كانت هناك حاجة إلى استخدام DLL التي تحتوي نوع مكتبة في أحد التطبيقات مثل أكسس فإنك قد تكون بحاجة إلى إضافة أحد المراجع إلى مكتبة النوع إذا لم يكن برنامج التطبيق قد قام بالفعل بإضافة المرجع. ويمكن وضع المرجع في مكتبة النوع عن طريق فتح الوحدة النمطية ثم اختيار Tools ⇦ References ويتم إضافة المرجع عن طريق اختيار المربع الموجود على اليسار من مكتبة النوع في القائمة. وإذا كانت مكتبة النوع لا تظهر في القائمة، انقر زر Browse لوضع المكتبة في ملفات النظام الخاصة بجهاز الحاسب الخاص بك. وبعد إضافة أحد المراجع يمكن استخدام الإجراء الموجود في DLL كما لو أنها كانت موجودة في داخل VBA.

ملاحظة من أحد أمثلة DLL التي تستخدم مكتبة النوع هي مكتبة Data Access Object، حيث أنها تحتوي على كل الإجراءات الخاصة باستخدام كائنات الوصول إلى البيانات. كما أن الملف DAO360.dll يحتوي على كل من DAO DLL ومكتبة النوع. وعندما تقوم بإعداد أكسس 2000 فإنه يتم إضافة المرجع إلى مكتبة النوع آلياً.

استخدام عبارة الإعلان

تكون هناك حاجة إلى إخبار VBA عن مكان المكتبة في كل مرة تريد فيها استخدام الإجراء في المكتبة عندما لا يحتوي DLL على مكتبة النوع. كما أنك أيضاً ستحتاج إلى إخبار VBA عن كيفية استدعاء البرنامج. وإذا كنت تريد استخدام إجراء DLL في الوحدة النمطية فإنه يجب وضع عبارة Declare في جزء Declaration الخاص بالوحدة النمطية. وتعطي هذه العبارة المعلومات التالية عن الإجراء:

- مجال الإعلان حيث أنه في الوحدة النمطية القياسية يمكن استخدام الكلمة الأساسية Public إذا كنت تريد إجراء DLL أن يكون متاحاً في كل الإجراءات والوحدات النمطية في المشروع أو الكلمة الأساسية Private إذا كانت هناك حاجة إلى استخدام إجراء DLL أن يكون متاحاً في إجراءات الوحدة النمطية. ويمكن استخدام الكلمة الأساسية Private في الوحدة النمطية الخاصة بالنموذج أو التقرير.
- الاسم الخاص بالإجراء في الوحدة النمطية. ويمكن إعطاء الاسم المخصص إلى الإجراء عن طريق استخدام الكلمة الأساسية Alias. وإذا لم تقم بوضع أحد الأسماء المستعارة فإنه يجب أن يكون الاسم الخاص بالإجراء الموجود في الوحدة النمطية مماثلاً لاسم الإجراء الموجود في DLL. وتستخدم هذه الأسماء المستعارة عندما يكون لهذا الإجراء أحد الأسماء التي تشتمل على حروف غير صالحة. أو يحتوي على أسماء تتماثل مع أسماء الكلمات الأساسية التي تخص VBA.
- اسم ومسار DLL، حيث يتم وضع اسم المكتبة في علامات اقتباس كما أنه لا يعتمد على حجم الخط.
- اسم الإجراء في DLL حيث يعتمد على حجم الخط.
- أنواع البيانات ورقمها في الوسائط، حيث يتوقع DLL أن يتم وضع الوسائط في ترتيب خاص كما أن لها حجماً معيناً. ومن أصعب الأشياء هي الإعلان عن الوسائط على نحو صحيح عند استخدام DLL في VBA. كما يجب أن يتم الإعلان عن VBA بطريقة صحيحة تماماً حتى يتم تمرير الوسائط عن طريق القيمة أو المرجع تماماً مثلما يتقبلها DLL.
- نوع البيانات الخاص بقيمة المرتجع إذا كان الإجراء هو أحد الوظائف.

كما أن الصيغة الخاصة بالعبارة Declare لإجراء الوظيفة في DLL هي:

```
[Public|Private] Declare Function procedurename Lib "libraryname"
[Alias "aliasname"][([arglist])][As type]
```

حيث:

- Procedurename هو الاسم الخاص الذي تستخدمه في الوحدة النمطية لتعريف الإجراء.
- تقوم العبارة Lib بتحديد DLL الذي يقوم باستضافة الإجراء.
- Aliasname هو اسم الإجراء في DLL.

كما أن الصيغة الخاصة بالإجراء الفرعي تكون مشابهة فيما عدا أن الكلمة الأساسية Function يتم استبدالها بالكلمة Sub كما أنه من غير الممكن الرجوع عن ذلك. وتحتوي الوسيطة الموجودة في *arglist* على الصيغة:

[Optional][ByVal|ByRef][ParamArray] varname[()][As type]

حيث *Vername* هو اسم المتغير ويكون *type* هو نوع بيانات المتغير.

تحذير يجب تمرير عدد الوسائط وأنواع البيانات الخاصة بالوسائط التي يتوقعها DLL. وإذا لم تقم بعمل ذلك فإن DLL قد يحاول الوصول إلى الذاكرة التي ليس له حق الوصول إليها، وتكون النتيجة هي خطأ General Protection، بمعنى آخر حدوث مشكلة لأكسس مع ضياع العمل.

يمكن بعد إعلان الوظيفة في جزء Declaration استدعاؤها في إجراء VBA.

استخدام API DLL الخاص بويندوز

تعد المكتبات التي يقدمها الويندوز من أفضل الأمثلة على DLL التي تحتاج إلى عبارات Declare وهي تسمى *Windows API* (واجهة برمجة التطبيق). وهي تتضمن على أكثر من ١٠٠٠ وظيفة مضمنة في ويندوز. ويقوم بحفظ أغلب التطبيقات في ثلاثة DLL كما يلي:

- يتضمن Kernel32.dll الوظائف الخاصة بإدارة الذاكرة وإدارة المهام والتعامل مع الموارد والعمليات المتعلقة بذلك.
- يتضمن Usert32.dll الوظائف المتعلقة بإدارة ويندوز مثل الوظائف الخاصة بالقوائم والرسائل وما إلى ذلك.
- يتضمن GDI32.dll الوظائف المتعلقة بعرض الرسوم مثل الخط والرسم وما إلى ذلك.
- إلا أن VBA لا يمكنه التعامل مع بيئة الويندوز وهو ما يتميز به API.



من أحد المشاكل التي تواجه المستخدم عند العمل مع وظائف *Windows API* هي أنها لا توفر الحماية من الأخطاء، حيث أنه يمكن التسبب في خطأ General Protection عن طريق إعلان إحدى وظائف API على نحو غير سليم. ويجب حفظ كل الكائنات قبل القيام باختبار إحدى وظائف API، حيث أنه عند حدوث خطأ General Protection لا يمكن حفظ أي شئ.

افترض أنك تريد بعض المعلومات عن كيفية إعداد واجهة المستخدم فإنك في هذه الحالة قد لا تستطيع استخدام VBA. إلا أنه هناك وظيفة في *Windows API* يمكنك استخدامها في ذلك

واسمها GetSystemMertrics. وهي توجد في User32 DLL وهي تحتاج إلى عـــدد صحيـــح كوسيطة كما أنها تعيد عدد صحيح طويل أيضاً. ويتم تحديدي المعلومات التـــي تريـــد أن تقـــوم باستعادتها باستخدام التعليمات البرمجية الرقمية. ويوضح الجدول ١٥-١ بعض الأمثلـــة علـــى المعلومات التي يمكن استعادتها والثوابت المناظرة والأرقام المساوية لها.

الجدول ١٥-١: الوسائط الخاصة بوظيفة GetSystemMertrics

معلومـــات عـــن واجهـــة الويندوز	الثابت	القيمة
عرض الشاشة	SM_CXSCREEN	صفر
ارتفاع الشاشة	SM_CYSCREEN	١
عرض المؤشر	SM_CXCURSOR	١٣
ارتفاع المؤشر	SM_CYCURSOR	١٤
هل يوجد ماوس	SM_MOUSEPRESENT	١٩
هل تم تبديل الزر الأيمن والأيسر للماوس؟	MS_SWAPBUTTON	٢٣

يتم إدخال عبارات Declare لوظيفة GetSystemMertrics مع عمل إجراء فرعي يســتخدم هذه الوظيفة. للاستعراض باستخدام API DLL. افتح أولاً وحدة نمطية جديدة اسمها basAPI مع إدخال عبارة Declare التالية في جزء Declaration:

```
Public Declare Function GetSystemMetrics Lib "user32" (ByVal nIndex As
Long) As Long
```

تم في هذا المثال استخدام نفس اسم الوظيفة في الوحدة النمطية كالاسم المستخدم في وظيفـــة المكتبة، مما يتيح حذف جزء الاسم المستعار من عبارة الإعلان.

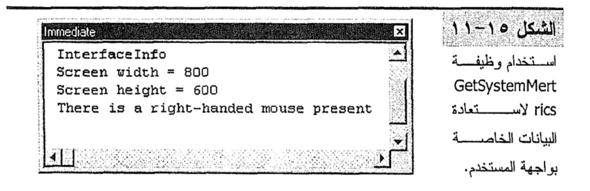
ثم قم بالإعلان عن الثوابت التالية في جزء Declaration:

```
Public Const SM_CXSCREEN = 0
Public Const SM_CYSCREEN = 1
Public Const SM_MOUSEPRESENT = 19
Public Const SM_SWAPBUTTON = 23
```

ثم قم بإدراج الإجراء InterfaceInfo الموضح فيما يلي:

```
Public Sub InterfaceInfo()
   Debug.Print "Screen width = " & GetSystemMetrics(SM_CXSCREEN)
   Debug.Print "Screen height = " & GetSystemMetrics(SM_CYSCREEN)
   If GetSystemMetrics(SM_MOUSEPRESENT) = 1 Then
      If GetSystemMetrics(SM_SWAPBUTTON) = 1 Then
         Debug.Print "There is a left-handed mouse present"
      Else
         Debug.Print "There is a right-handed mouse present"
      End If
   Else
      Debug.Print "There is no mouse present"
   End If
End Sub
```

اكتب **InterfaceInfo** مع الضغط على Enter لاختبار الإجراء في الإطار الحالي. يوضح الشكل ١٥-١١ النتائج الخاصة بأحد أجهزة الحاسب الذي يكون فيه حجم الشاشة ٨٠٠X٦٠٠ والماوس ذو الزر الأيمن.



الشكل ١٥-١١ استخدام وظيفة GetSystemMetrics لاستعادة البيانات الخاصة بواجهة المستخدم.

يتميز هذا المثال بالبساطة لأن وظيفة GetSystemMertrics تحتاج إلى وسيطة واحدة فقط.

استخدام ActiveX

يعد ActiveX أو OLE كما كان يسمى من قبل هو التقنية التي توفرها مايكروسوفت للسماح للتطبيقات بالاتصال مع بعضها البعض. وهو عبارة عن مجموعة من المعلومات على هيئة

كائنات. ويحتوي على عنصرين أساسيين وهما: نموذج الكائن المسمى COM ومجموعة من الإمكانات الخاصة به. وتتضمن هذه المجموعة:

OLE: وعن طريق استخدامه فإنه من الممكن أن يتم وضع أحد الكائنات التي تم وضعها في أحد التطبيقات في أحد المستندات الأخرى في تطبيق آخر. وهناك طريقتان لتخزين الكائنات: وهما الارتباط والتضمين. وعند ربط أحد الكائنات فإنك تقوم بتخزين نسخة من بياناته وارتباط مع الملف الأصلي. وإذا قمت بعمل بعض التغييرات على الملف الأصلي فإنها تنتقل إلى الكائنات المنضمة. وعند تضمين أحد الكائنات فإنه يتم تخزين نسخة فقط من بيانات الكائن بدون الارتباط بالملف الأصلي. وعند تضمين أحد الكائنات لا يكون هناك ارتباط بالملف الأصلي مما ينتج عنه أن التغييرات التي تطرأ على الملف الأصلي لا يتم نقلها إلى الكائن المضمن.

التنشيط الموضعي: ويمكن عن طريقة تحرير أحد الكائنات المضمنة في أحد عناصر التحكم من داخله. وللقيام بذلك انقر عنصر التحكم نقرة مزدوجة لتنشيطه. ويؤدي تنشيط الكائن المضمن إلى الالتحام بين التطبيق الأصلي والتطبيق الخاص بالمستند. فمثلاً عند تنشيط مستند مضمن في وورد في أحد نماذج أكسس فإن القوائم وأشرطة الأدوات الخاصة بتطبيقات وورد تستبدل تلك الخاصة بأكسس. كما يمكن استخدامها بحرية. وتكون هذه التقنية متاحة فقط في الكائنات المضمنة فقط.

التنفيذ الآلي: ويمكنك باستخدامه كتابة الإجراءات التي تقوم بإرسال التعليمات إلى أحد التطبيقات الأخرى والتحكم في الكائنات الأخرى التي يمكن أن تقوم الكائنات الأخرى بتنفيذها. وتوفر هذه الميزة التعامل مع الكائنات الموجودة في تطبيقات أخرى. ويتم التعرض إلى هذه التقنية فيما بعد بمزيد من التفصيل.

عناصر تحكم ActiveX: ويمكن إضافته إلى أحد المستندات في أحد التطبيقات. ويكون للكائن خصائصه وطرقه المميزة له كما أنه يستطيع معرفة الأحداث الخاصة به. وبعد إضافة **ActiveX** إلى عنصر التحكم فإنه يمكن بعد ذلك إرسال التعليمات إلى عنصر التحكم. ويتم التعرض إلى استخدام **ActiveX** في الجزئية التالية.

ملاحظة تختلف تطبيقات ويندوز بشكل واضح من حيث الاستجابة مع **ActiveX**. كما أن هذه التطبيقات تختلف في قدرتها على التعامل مع عناصر التحكم الخاصة به.

يستطيع أكسس ٢٠٠٠ تضمين وربط الكائنات التي تم إنشاءها في تطبيقات أخرى. وتتوقف إمكانية استخدام التنشيط الموضعي على ما إذا كان التطبيق الذي قام بإنشاء الكائن يتوافق مع ActiveX. ويمكن استخدام VBA للتحكم في كائنات عناصر التحكم التي تخص تطبيقات أخرى تتوافق مع Automation. كما يمكن أيضاً التحكم في الكائنات في أكسس ٢٠٠٠ من تطبيق آخر يتوافق مع المعايير القياسية التي تخص Automation.

استخدام عناصر تحكم ActiveX

تضفي عناصر تحكم ActiveX على التطبيق المزيد من القدرات المفيدة والفعالة بدون الحاجة إلى كتابة تعليمات البرمجة. ويتم اختبار عناصر التحكم وتصحيحها وإعدادها للاستخدام. وتتضمن الأمثلة عليها أزرار الحركة وأشرطة التقدم.

وتحتوي على الأحداث والطرق والخصائص الخاصة بها، وبالإضافة إلى ذلك فإنه عند إدراج عنصر تحكم ActiveX في نموذج لأكسس فإنك بذلك تقوم بوضعه في إطار أكسس وكنتيجة لذلك فإن أحداث وطرق وخصائص أكسس تنطبق أيضاً على عناصر تحكم ActiveX.

مكان عناصر تحكم ActiveX

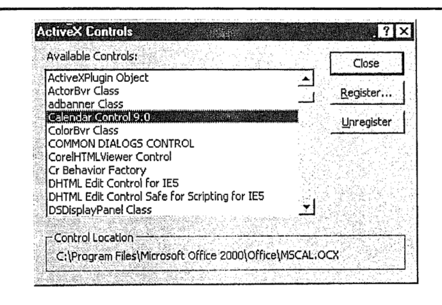
منذ زمن ليس بالقصير كانت عناصر تحكم ActiveX جزءاً من تقنيات مايكروسوفت للاتصال بين المكونات البرمجية تحت اسم عناصر تحكم OLE أو ocx.. إلا أن وجود الشبكات الضخمة قد أدى إلى ظهور عناصر تحكم ActiveX مع التأكيد على البرامج الصغيرة المعدلة. ودائماً ما تقدم صفحات الإنترنت إمكانية تحميل عناصر تحكم ActiveX، حيث أنه يوجد منها ما يزيد على ١٠٠٠ نوع في الأسواق.

ويمكن الحصول على عناصر التحكم المخصصة من مايكروسوفت أو من بعض الوكلاء المختصين بذلك. ويتضمن أكسس ٢٠٠٠عنصر تحكم مخصص واحد فقط لمجرد البداية وهو Calendar. كما أن العديد من عناصر التحكم التي تصاحب برنامج فيجوال بيسيك ٦ يمكن أن تعمل مع أكسس. ويمكن الحصول على قائمة ببائعي عناصر تحكم ActiveX الخاصة بأكسس من http://www.microsoft.com/accessdev. كما يمكن أيضاً إنشاء عناصر تحكم ActiveX الخاصة بك باستخدام عدد غير قليل من لغات البرمجة بما في ذلك فيجوال بيسيك.

تثبيت وتسجيل عنصر تحكم ActiveX

يجب أن يكون لعنصر تحكم ActiveX الإدخالات الصحيحة في Windows Registery قبل إمكانية استخدامه. إلا أن بعض عناصر التحكم تحتوي على برامج تثبيت تقوم بتسجيل وتثبيت عنصر التحكم بشكل آلي، والبعض الآخر يجب تثبيته يدوياً.

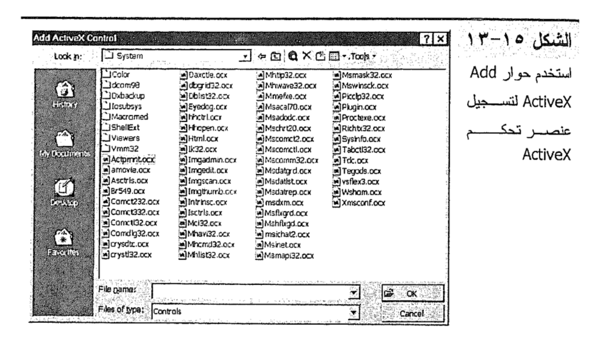
وعند تثبيت وتسجيل عنصر تحكم ActiveX فإن مكتبة النوع الخاصة بها يتم تثبيته آلياً. وتتيح لك هذه الميزة استخدام حوار ActiveX Control "انظر الشكل ١٥-١٢". وتحتوي عناصر تحكم ActiveX على الملحق ocx.. كما أنه يوجد في أكسس ٢٠٠٠ عنصر تحكم Calendar لذلك فإنه يجب رؤية عناصر التحكم من القائمة.



الشكل ١٥-١٢ حوار ActiveX Control عناصر التحكم المسجلة في Windows Registery على جهاز الحاسب الآلي.

لتسجيل أحد عناصر التحكم الجديدة انقر زر Register من حوار ActiveX Control. حدد مكان ملف عنصر التحكم في حوار Add ActiveX Control "انظر الشكل ١٥-١٣". ولحذف تسجيل أحد عناصر التحكم انقر زر Unregister من حوار ActiveX Control.

وسوف يتم على سبيل المثال استخدام عنصر تحكم Calendar الموجود مع أكسس ٢٠٠٠ وهو يحتوي على خصائص معينة يتم استخدامها لتحديد شكل عنصر التحكم وتعيين البيانات وقراءتها في عنصر التحكم. ويحتوي عنصر التحكم على طرقاً معينة تستخدم لضبط اليوم والشهر والسنة ولتحديث التوقيم. كما أن عنصر التحكم يحتوي على طريقة معينة لعرض مربع About مع معلومات الإصدار وحقوق الطبع. كما أن عنصر التحكم يقوم بتنظيم الأحداث الخاصة بعنصر التحكم عندما يقوم المستخدم بنقره، أو التغيير إلى تاريخ جديد أو الضغط على أحد المفاتيح. كما أن عنصر التحكم يستوعب الحدث عند تغيير التاريخ.



الشكل ١٥-١٣

استخدم حوار Add ActiveX لتسجيل عنصر تحكم ActiveX

إدراج عنصر التحكم ActiveX

هناك طريقتين لوضع عنصر التحكم في النموذج بعد تثبيت وتسجيل عنصر تحكم ActiveX: وهما استخدام أمر القائمة أو زر More Control في مربع الأدوات.

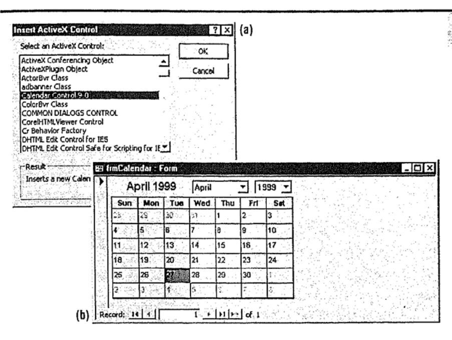
استخدام قائمة الإدراج

يمكن وضع عنصر تحكم ActiveX باستخدام قائمة Insert. تتبع التعليمات التالية لإدراج عنصر تحكم Calendar:

١- افتح نموذج جديد من عرض Design مع تسميته frmCalendar. اختر Insert ⇦ ActiveX Control، فيظهر حوار Insert ActiveX Control" انظر الشكل ١٥-١٤ أ".

٢- اختر Calendar Control 9.1 مع نقر OK. فيتم إدراج عنصر التحكم Calendar وعرض البيانات الحالية عندما يكون النموذج في عرض Design.

٣- انتقل إلى عرض Form" انظر الشكل ١٥-١٤ ب". اختر الشهر والسنة باستخدام مربعات التحرير والسرد ثم انقر على مربع اليوم لاختيار اليوم في الشهر المعروض. "يتم في الجزئية التالية التعرض لاستخدام التاريخ المختار."



الشكل ١٥-١٤

اختر أمر ActiveX Control في قائمة Insert لعرض قائمة بعناصر التحكم المسجلة "أ". اختر الشهر والسنة باستخدام مربعات التحرير والسرد من عنصر التحكم Calendar ثم انقر على مربع اليوم لاختيار اليوم "ب".

٤ – انتقل إلى عرض Design مع حذف عنصر التحكم Calendar.

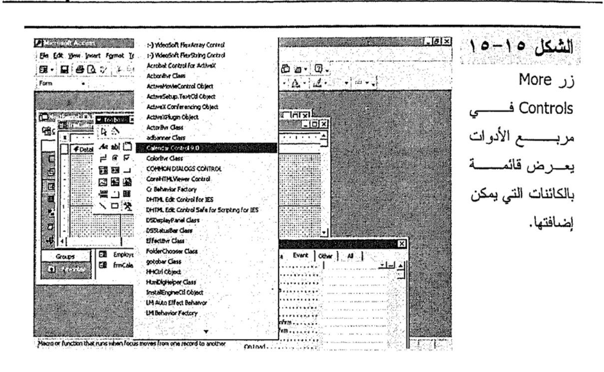
استخدام زر المزيد من عناصر التحكم

يعد إضافة زر إلى عنصر التحكم ActiveX من الطرق الأخرى لتحديد وضعه. وبعد ذلك يتم اختيار عنصر التحكم من مربع الأدوات مع سحبه إلى النموذج.

انقر زر More Controls في مربع الأدوات ثم اختر Calendar Control 9.0 من مربع قائمة الكائنات لإضافة عنصر التحكم Calendar "انظر الشكل ١٥-١٥"، ثم انقر النموذج لإدراج عنصر التحكم Calendar.

عرض مراجع عنصر التحكم

عند وضع عنصر التحكم ActiveX على النموذج باستخدام أياً من هذه التقنيات التي تم التعرض لها فإن ذلك يؤدي إلى حدوث شيئين: يتم وضع عنصر التحكم في إطار الكائن غير المترابط، ويقوم أكسس آلياً بإضافة أحد المراجع إلى مكتبة نوع عنصر التحكم في التطبيق.



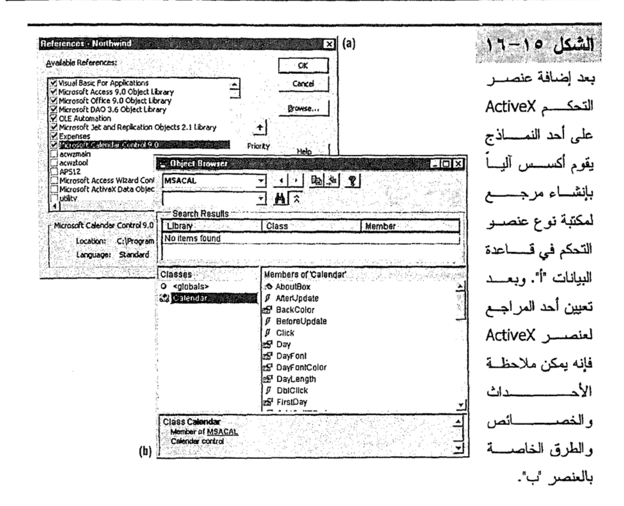
الشكل ١٥-١٥

زر More Controls في مربع الأدوات يعرض قائمة بالكائنات التي يمكن إضافتها.

ولمشاهدة المرجع افتح الوحدة النمطية ثم اختر Tools ⇦ References لفتح حوار References "انظر الشكل ١٥-١٦أ". ولعرض قائمة بخصائص وطرق وأحداث عناصر التحكم افتح Object Browser ثم ضع قائمة عناصر التحكم في مربع التحرير والسرد Project/Library. اختر عنصر التحكم Calendar لعرض قائمة بخصائص وطرق وأحداث عناصر التحكم "انظر الشكل ١٥-١٦ب".

ملاحظة

يحتوي عنصر التحكم ActiveX على طرق SetFocus وsizeToFit الخاصة بأكسس. لاحظ أن Object Browser يقوم بذكر خصائص وطرق وأحداث عناصر التحكم ActiveX فقط. إلا أنها لا تقوم بذكر خصائص وطرق وأحداث أكسس التي تنطبق على عناصر التحكم ActiveX المدرجة.



الشكل ١٥-١٦ بعد إضافة عنصر التحكم ActiveX على أحد النماذج يقوم أكسس آلياً بإنشاء مرجع لمكتبة نوع عنصر التحكم في قاعدة البيانات "أ". وبعد تعيين أحد المراجع لعنصر ActiveX فإنه يمكن ملاحظة الأحداث والخصائص والطرق الخاصة بالعنصر "ب".

تعيين خصائص عناصر تحكم ActiveX

يحتوي عنصر تحكم ActiveX على خصائص أكسس الموضحة فيما يلي. ويمكن تعيين بعض الخصائص لعنصر تحكم ActiveX في وضع Design بينما يتم تعيين الخصائص الأخرى في الماكرو أو الإجراءات.

Class	BorderWidth	BorderStyle	BordercOLOR
Enable	DisplayWhen	ControlTipText	ControlSource
Locked	Left	Height	HelpContextID
OnEnter	OLEClass	Object	Name
OnUpdated	OnLostFocus	OnGotFocus	OnExit
Tag	TabStop	TabIndex	SpecialEffect
Width	Visible	Verb	Top

ملاحظة تكون الخصائص ControlSource و locked متاحة فقط لعناصر تحكم ActiveX التي يمكن أن تكون مرتبطة بحقل الجدول.

وقد تكون خصائص عنصر التحكم التي قد قمت بتعيينها في عرض Design مذكورة في ورقة خصائص أكسس حسب نوع عنصر التحكم، كما أنه من الممكن أن يتم ذكر الخصائص الخاصة بعنصر التحكم في حوار الخصائص لعنصر التحكم ActiveX.

وهناك طريقتان لتعيين خصائص عنصر التحكم عند العمل مع عنصر التحكم ActiveX في عرض Design. ويمكن استخدام ورقة خاصية عنصر التحكم، أو إذا ما كان عنصر التحكم له حوار مخصص للخاصية فإنه يمكن استخدامه.

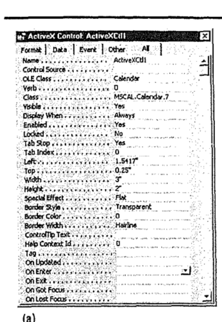
استخدام ورقة الخاصية

يتم وضع عنصر التحكم المخصص في إطار الكائن غير المترابط، لذلك فإنه عند اختيار عنصر التحكم المخصص في عرض Design فإن ورقة الخاصية تقوم بعرض خصائص أكسس للإطار، بالإضافة إلى الأحداث الخاصة بعنصر التحكم. وتتضمن ورقة خاصية Calendar Control كل من خصائص إطار الكائن "انظر الشكل ١٥-١٧ أ" والخصائص الخاصة بالعنصر بدءاً بخاصية Custom "انظر الشكل ١٥-١٧ ب".

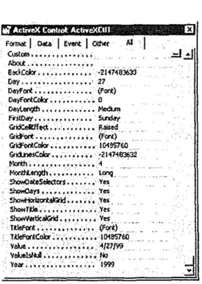
ولاستكشاف الخصائص الخاصة بالعنصر انتقل إلى عرض View مع فتح الإطار الحالي:

- اكتب ?Forms!frmCalendar!ActiveXCtl1.Value. ثم اضغط على Enter. فيقوم الإطار الحالي بإظهار البيانات الحالية. ويمكن تعيين تاريخ التقويم عندما يكون النموذج في عرض Design أو Form.
- اكتب Forms!frmCalendar!ActiveXCtl1.Value = #6/6/99#. ثم اضغط على Enter. وبعد ذلك فإن التقويم يتغير ليعرض التاريخ. يمكن تغيير شكل التقويم عن طريق تغيير الخصائص.
- اكتب Forms!frmCalendar!ActiveXCtl1.ShowTitle = False ثم اضغط على Enter. فيتم حذف عنوان عنصر التحكم Calendar.

الشكل ١٥-١٧ تذكر ورقة نمط أكسس لعنصر تحكم Calendar الخصائص التي يمكن تعيينها في وقت Design بما في ذلك خواص إطار الكائن "أ" وخصائص عنصر التحكم التي تبدأ بخاصية Custom "ب". إلا أن ورقة الخاصية لا تتضمن الأحداث الخاصة بعناصر التحكم.



(a)



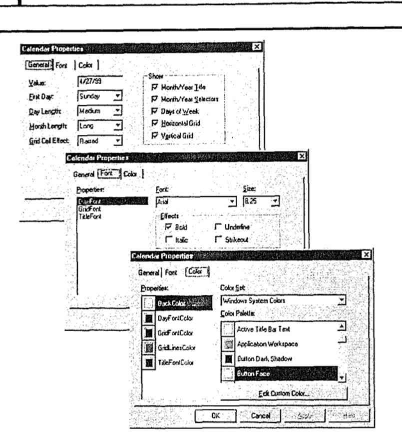
(b)

استخدام حوار الخاصية المخصصة

يمكن عرض حوار الخاصية المخصصة لعنصر تحكم ActiveX عن طريق استخدام أحد التقنيات التالية:

- انقر زر Build التالي لخاصية Custom المذكورة في ورقة الخاصية.
- انقر نقرة يمنى على عنصر التحكم ثم اختر الأمر Calendar Object ثم الأمر Properties من قائمة الاختصارات.
- انقر نقرة مزدوجة على عنصر تحكم ActiveX.

ويحتوي حوار خصائص عنصر التحكم على علامات تبويب لتعيين الخطوط والألوان وما إلى ذلك. "انظر الشكل ١٥-١٨".



الشكل ١٥-١٨

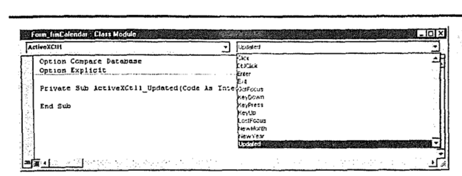
يحتوي حوار Calendar Properties على علامات التبويب التي تستخدم في التحكم في الخصائص التي تتعامل مع عناصر التحكم المعينة في وقت Design بما في ذلك "أ" وهي العامة و"ب" الخط و"ج" اللون.

استخدام الأحداث

تحتوي الأحداث التي يستطيع عنصر التحكم التعامل معها على خصائص مناظرة يتم ذكرها في فئة Event لورقة الخاصية. كما يمكن استخدام الحدث لعناصر التحكم المضمنة لبدء الماكرو، أو أحد إجراءات الأحداث. وتكون الأحداث الخاصة بعناصر التحكم في الوحدة النمطية الخاصة بالنموذج عن طريق اختيار View ⇦ Code أو نقر زر Code من شريط الأدوات عندما يكون النموذج في عرض Design.

ملاحظة يمكن بدء إجراءات الأحداث وليس إجراءات الوظائف أو الماكرو باستخدام حدث عنصر التحكم ActiveX.

اختر عنصر التحكم Calendar من عرض Design ثم انقر زر Code من شريط الأدوات. اختر ActiveXCtl1 في مربع السرد والتحرير Object على يسار لعرض قالب التعليمات البرمجية لحدث Updated انقر مربع التحرير والسرد الموجود على اليمين لعرض قائمة بكل من أحداث أكسس التي يستطيع إطار عنصر التحكم والأحداث الخاصة بعنصر التحكم.



الشكل ١٥-١٩

يذكر مربع التحرير والسرد Procedure لعنصر التحكم ActiveX كل الأحداث لعنصر التحكم، بما في ذلك كل من أحداث إطار الكائن والأحداث الخاصة بعنصر التحكم.

سوف نقوم بإضافة أحد التقويمات إلى نموذج Orders كأحد الأمثلة على استخدام عنصر التحكم Calendar مع ربطها بعنصر التحكم Required Data مما يتيح استخدام عنصر التحكم Calendar لاختيار البيانات:

١- افتح نموذج Orders من عرض Design مع سحب الحد السفلي من جزء Detail لعمل مساحة للتقويم.

٢- انقر More Controls في شريط الأدوات ثم اختر عنصر التحكم Calendar انقر RequiredDate من قائمة الحقل ثم اسحب إلى الجزء السفلي من النموذج. قم بتغيير خاصية Name إلى ActiveXCalendar مع تغيير خاصية ShowTitle إلى No. مع تعيين حجم عنصر تحكم التقويم. ألان أصبح عنصر تحكم Calendar منضماً إلى حقل RequiredDate . أضف التسمية إلى مع خاصية Caption بعد ضبطها على Select Required Date.

٣- احفظ النموذج ثم انتقل إلى عرض View. استعرض في السجلات. ويقوم Calendar بعملية التحديث آلياً لعرض قيمة حقل RequiredDate للسجل" انظر الشكل ١٥-٢٠".

الشكل ١٥-٢٠

عنصر التحكم Calendar منضماً إلى حقل RequiredDate مع التحديث آلياً عند استعراض سجل آخر.

٤- غير التاريخ المطلوب لأحد الأوامر باستخدام عنصر التحكم Calendar. ولا يقوم عنصر التحكم RequiredDate بعملية التحديث آلياً إلا أنه يمكن تحديث عنصر التحكم عن طريق اختيار Records ⇦ Refresh أو عن طريق استخدام F9.

٥- لتحديث النموذج آلياً يمكن إنشاء أحد إجراءات الأحداث لحدث AfterUpdate من عنصر التحكم Calendar. انتقل إلى عرض Design اختر عنصر التحكم Calendar ثم انقر زر Code في شريط الأدوات مع ادخال إجراء الحدث التالي:

```
Private Sub ActiveXCalendar_AfterUpdate
    Me.Refresh
End Sub
```

٦- احفظ الوحدة النمطية ثم انتقل إلى عرض Design. غير التاريخ المطلوب باستخدام عنصر التحكم Calendar لاحظ أن النموذج يتغير آلياً.

استخدام الحركة

وهي من أحد أهم المميزات في تقنية ActiveX للاتصال بين المكونات البرمجية. ويمكن باستخدامها يمكن عمل العديد من التطبيقات التي تحتوي على العديد من المكونات القوية المتميزة.

فهم الحركة

تعتبر الحركة هي العملي التي يقوم بها التطبيق الذي تتعامل معه بإرسال التعليمات إلى تطبيق آخر. وتسمى التطبيقات التي يمكن أن تستخدم كائنات من تطبيق آخر أو التي تتيح استخدام الكائنات الخاصة بها مكونات *ActiveX*. ويسمى التطبيق الذي يقوم بتوفير هذه الكائنات خادم COM. كما يسمى التطبيق الذي تقوم بكتابة الإجراءات فيه COM Cntroller.كما تسمى الكائنات التي يتم التحكم فيها بكائنات *ActiveX*.

ويعتبر كل من تطبيقات أوفيس هي مكون *ActiveX*. وتظهر إمكانيات أكسس في تخزين البيانات واستعاداتها وتحديثها. وورد ٢٠٠٠ في تنسيق وتقرير البيانات وإكسل في تحليل البيانات...الخ. وتوفر كل من هذه التطبيقات مجموعة من كائنات ActiveX مما يمكن استخدامها في العديد من المهام. إلا أن الميزة الحقيقة تكمن في إمكانية بناء المميزات الأساسية للعديد من التطبيقات التي تمت كتابتها ومعرفتها على نحو دقيق واستخدامها في المشروع الخاص بك.

وفي بعض الأحيان تكون بعض التطبيقات خادم COM فقط أو متحكمات COM فقط. كما أن بعض التطبيقات يمكنها التعامل بالطريقتين مثل أكسس ٢٠٠٠ وأكسل ٢٠٠٠ و2000 Project وفيجوال بيسيك ٦.

ملاحظة يتميز ActiveX بكونه يقبل التعامل مع أكثر من تقنية واحدة حيث أن الإصدارات السابقة كانت ذات قدرة محدودة من هذه الناحية.

وعندما يقوم أحد التطبيقات بتوفير كائنات ActiveX وتكون المخططات أو برامج العمل متاحة فقط، التي تسمى بلغة البرمجة فئات أو أنواع الكائن. كما تسمى الكائنات التي تقوم بإنشائها بالأمثلة. وتشتمل الفئات على الاسم وتعريف الخاصية أو الطريقة والأحداث التي يستوعبها الكائن. ويقوم خادم COM بتخزين كل البيانات عن الكائنات الخاصة به. وتحتوي المكتبة الخاصة بالكائن على الملحق olb. أو الملحق tlb.. كما أن ملف التعليمات له الملحق hlp.. وفيما يلي بعض الأمثلة على ذلك:

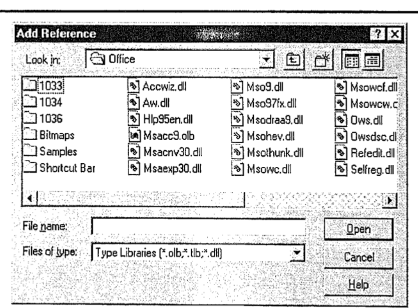
- يتم توفير أنواع كائنات أكسس في MSAcc9.olb مع المساعدة من Acmain90 وكل منهما يتم تثبيته آلياً في Program Files\Microsoft Office\ Office folder\.
- يتم توفير الكائن DAO لقاعدة البيانات Jet في DAO360.dll والذي يتم تثبيته آلياً عند تثبيت أكسس ٢٠٠٠ أو إكسل ٢٠٠٠ أو فيجوال بيسيك ٦ أو فيجوال ++C. والمجلد

الافتراضي هو either \Windows\System or \Program Files\Common Files\Microsoft Shared\DAO

- يتم توفير كائن إكسل في ملف Excel9.olb مع التعليمات في ملف Xlmain9 كما أن كل من الملفات يتم تثبيته في \Program Files\Microsoft Office\ Office folder.
- يتم توفير النموذج VBA في الملف VBA332.dll في \Program Files\Common Files\Microsoft Shared\VBA folder. كما أن ملفات التعليمات تكون في \Program Files\Microsoft Office\ Office folder. كما أن ملف Help لتطبيقات وأوفيس الأخرى يوجد في \Program Files\Common Files\Microsoft Shared\VBA folder.

كائنات التطبيق

ويمكن القيام بهذه العملية من أكسس عن طريق إضافة مكتبة كائن التطبيق إلى Object Browser. وكما هو موضح فيما سبق في هذا الفصل فإنه للقيام بهذه العملية افتح أي وحدة نمطية في المشروع ثم اختر Tools ⇦ Refrences وإذا لم يكن أحد العناصر غير مذكور في حوار Reference فإنه يمكن نقر زر Browse لعرض حوار Add Reference. حيث يمكن منه تحديد موضع مكتبة النوع على الجهاز "انظر الشكل ١٥–٢١" ثم انقر OK لإضافة المرجع.



الشكل ١٥–٢١

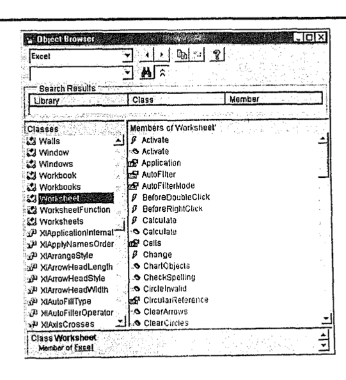
إذا كانت مكتبة الكائن لا توجد في قائمة حوار Reference انقر زر Browse لعرض حوار Add Reference وتحديد موضع مكتبة الكائن.

عند إضافة أحد المراجع إلى مكتبة النوع فإنك تقوم بتخزين مسار مكتبة النوع في مشروعك.
وإذا ما قمت بنقل مكتبة النوع فإنك سوف تكون بحاجة لإنشاء مرجع جديد إلى هذه المكتبة الذي

يتم إضافته إلى مشروع أكسس كما أن VBA يستطيع أيضاً أن يتعرف عليه في المكتبة ثم يمكن بعد ذلك استخدامها بنفس الطريقة التي يتم بها استخدام كائنات أكسس وDAO.

ملاحظة يفترض هذا الفصل أنك قد قمت بالفعل بتثبيت أوفيس ٢٠٠٠ على جهازك. وإذا لم تكن قد قمت بذلك يمكن إضافة مرجع إلى خادم COM آخر الذي تمتلكه. أما إذا كنت تمتلك أكسس على الجهاز الخاص بك فإنه يمكنك تعديل بعض الأمثلة واستخدام أكسس على أساس كل من خادم والمتحكم في COM.

يمكنك استخدام Object Browser بعد إضافة المرجع لاستعراض كائنات التطبيق للمزيد من المعلومات ارجع إلى الفصل ٥. يوضح الشكل ١٥-٢٢ خصائص وطرق كائن Worksheet في Excel 9.0 Object Library. وعند إضافة المرجع فإنك أيضاً تقوم بإضافة تعليمات فورية لكائنات التطبيق.



الشكل ١٥-٢٢
يمكن استخدام Object Browser في أكسس لاستعراض الكائنات في إكسل.

قبل أن تقوم باستخدام Automation على نحو ناجح فإنه يجب أن يتم معرفة نوع الكائن الذي تريد التعامل معه. وفي واقع الأمر فإن الكائنات الأخرى التي توجد في التطبيقات الأخرى قد تكون أكثر تعقيداً كما أن أنواع الكائنات التي تخص ActiveX تتغير من إصدار لآخر.

استخدام المعلومات في مكتبة الكائن

تعتبر الميزة الحقيقية من إضافة أحد المراجع إلى مكتبة الكائن هي أن VBA يستطيع أن يستخدم المعلومات في مكتبة الكائن حيث أنه يقوم بتحميل مكتبة الكائن في الذاكرة مما يتيح إمكانية البحث عن المعلومات المطلوبة عند تجميع الوحدة النمطية. ويمكنك عند كتابة الإجراءات للتعامل مع الكائنات فإنه يمكن استخدام المعلومات الموجودة في هذه المكتبة والتي يستخدمها التطبيق الآخر. فمثلاً إذا كنت تعمل مع إكسل يمكن استخدام الثابت مع الملحق xl. في إجراءات أكسس ولن تكون بحاجة إلى استخدام الرقم المناظر أو إعادة تعريف هذه الثوابت.

كما يمكن أيضاً إعلان المتغيرات لنوع معين من البيانات. فمثلاً إذا كنت تتعامل مع إكسل فإنه يمكنك إعلان متغيرات المخطط وورقة العمل وما إلى ذلك.

ملاحظة تسمى العملية التي يتم من خلالها التأكد من وجود الكائن وكون إحدى الطرق أو الخصائص صحيحة بالتوثيق. ويمكن استخدام هذه الطريقة في حالتين اثنتين: في أثناء وقت التجميع أو في وقت التشغيل. وعند الإعلان عن أحد المتغيرات كنوع بيانات معين فإنك في هذه الحالة تقوم باستخدام التوثيق المبكر أما عند إعلان المتغير من نوع بيانات Object فإنك في هذه الحالة تستخدم التوثيق المتأخر.

ودائماً ما تحتوي التطبيقات على نفس أنواع البيانات مع نفس الأسماء. فمثلاً تحتوي كل تطبيقات أوفيس على كائن Application. كما يمكن تهيئة نوع البيانات باستخدام اسم التطبيق مع الصيغة application.objecttype. ولتحديد الاسم الذي يستخدم لتحديد التطبيق في Automation افتح Object Browser ثم ابحث عن الاسم في القائمة المعروضة في مربع التحرير والسرد Project/Library. وعند الإعلان عن متغير لكائن Automation فإنه يمكن إعداد نوعن بيانات الكائن لمنع VBA من إنشاء الكائن الخاطئ. انظر العبارة التالية التي تستخدم التوثيق المبكر للإعلان عن متغيرات الكائن:

```
Dim appAccess As Access.Application
Dim appExcel As Excel.Application
Dim wine As Excel.Window
Dim winp As Project.Window
```

ملاحظة ليست هناك حاجة إلى إضافة المرجع إلى مكتبة كائن التطبيق لاستخدام كائن Automation، إلا أنه عند عدم إضافته فإنك لن تستطيع استخدام أياً من ثوابت التطبيق أو أنواع بيانات معينة، حيث أن استخدام أياً منهما يؤدي إلى حدوث خطأ وقت التشغيل.

استخدام كائنات Automation

هناك أربعة خطوات أساسية للتحكم في كائن Automation في الإجراء:

١- قم بإنشاء متغير الكائن لكائن Automation.

٢- قم بإنشاء ثابت لكائن Automation مع تعيين متغير الكائن على الثابت الجديد.

٣- تحكم في الكائن في الكائن عن طريق تعيين خصائصه أو الحصول عليها أو عن طريق تشغيل الطرق الخاصة به.

٤- اغلق الكائن عند الانتهاء.

عند الحاجة إلى استخدام كائنات Automation للتطبيق فإنه يجب أولاً الوصول إلى الكائن وأول خطوة هي عمل أحد الأمثلة على أحد الكائنات في التطبيق.

وكما هو موضح في الفصل السادس فإنه عند استخدام أحد كائنات الوصول فإنه يجب البدء من قمة تسلسل كائنات DAO مع كائن DBEngine مع اجتياز الهيكل الهرمي إلى الكائن المطلوب. وبالطريقة نفسها فإنه عند الرغبة في التعامل مع كائن Automation من أحد التطبيقات الأخرى فإنه يجب البداية من أحد الكائنات التي يستطيع نظام التشغيل التعرف عليها. ودائماً ما تبدأ بالكائن الذي يحتل أعلى قمة الهيكل الهرمي وهو الذي دائماً ما يكون كائن Application.

وعند الشروع في عمل أحد الأمثلة الجديدة في كائن Application فإن نظام التشغيل يبدأ في عمل مثال للتطبيق. ويمكن عند بداية التطبيق اجتياز الهيكل الهرمي إلى الكائن الذي تريد التعامل معه. إلا أن بعض التطبيقات تحتوي على أكثر من كائن يستطيع نظام التشغيل التعرف عليه. فمثلاً يستطيع ويندوز التعرف على كائنات Workbook وChart بالإضافة إلى كائن Application لإكسل. وفي هذه الحالة فإنه عند إنشاء أحد الأمثلة الجديدة الذي يشير إلى Workbook أوChart فإن ويندوز يبدأ أحد أمثلة إكسل ثم يقوم بإنشاء أحد الأمثلة من كائنات Workbook أوChart. ثم يتم الانتقال إلى الكائن المطلوب التعامل معه.

هناك ثلاثة طرق لبناء مثال لكائن Automation: عن طريق استخدام الوظيفة CreateObject. أو عن طريق استخدام الوظيفة GetObject أو عن طريق استخدام الكلمة

الأساسية New. وفي أي من هذه الطرق يقوم خادم COM ببدء الإطار المخفي بنفس الطريقة ويظل مخفياً حتى تقوم أنت بجعله مرئياً. وفي بعض الأحيان فإنك تحتاج إلى إمكانات خادم COM كما أنك قد لا تحتاج إلى جعل الكائن مرئياً. فمثلاً عند استخدام الوظائف الرياضية لتطبيق إكسل فإنك لا تحتاج إلى إظهار إطار إكسل.

تلميح يحتاج عرض العروض التقديمية إلى المزيد من الوقت، لذلك فإنه من غير المستحسن عرض كائن Automation إلا إذا كان هناك حاجة إلى ذلك بالفعل.

استخدام وظيفة CreateObject

وتستخدم لإنشاء أحد الأمثلة الجديدة من أحد الكائنات التي تخص Automation التي يستطيع نظام التشغيل التعرف عليها. ثم يقوم باستعادة أحد المراجع إلى الكائن. والصيغة الخاصة بوظيفة CreateObject هي:

```
Set objvar = CreateObject (appname.objecttype)
```

وobjvar هو أحد متغيرات الكائن، وappname هو اسم التطبيق كما هو مذكور في Object Browser،وobjecttype هي اسم نوع الكائن المراد إنشاؤه. فمثلاً عند العمل مع وورد قم بالإعلان عن متغير الكائن ثم قم بإنشاء الكائن المراد الإشارة إليه كما يلي:

```
Dim appWord As Word.Application
Set appWord = CreateObject("Word.Application")
```

وتقوم وظيفة CreateObject ببدء مثال على تطبيق خادم COM. وإذا كان الخادم يقوم بالفعل بتشغيل أحدها، فإن CreateObject تقوم ببدء مثال آخر، إلا إذا كان التطبيق لا يسمح إلا لمثال واحد فقط. وفي هذه الحالة يتم إنشاء مثال واحد فقط بغض النظر عن عدد المرات التي يتم فيها تشغيل وظيفة CreateObject.

ولتوضيح هذه الفكرة فإن إجراء LaunchWord الموضح فيما يلي يستخدم وظيفة CreateObject لإنشاء مثال على وورد. في هذه الحالة يمكن الوصول إلى الوحدة النمطية لكائن وورد عند تشغيله كما يمكن استخدام طريقة Open لكائن Document لفتح عينة مستند Product.doc.

```
Public Sub LaunchWord()
    Dim appWord As Word.Application, strdoc As String
    Set appWord = CreateObject ("Word.Application")
```

```
    MsgBox "Word is running"
    appWord.Visible = True
    strdoc = "c:\Program Files\Microsoft Office\Office\"
    strdoc = strdoc & "Samples\Products.doc"
    appWord.Documents.Open (strdoc)
    Set appWord = Nothing
End Sub
```

وبعد الانتهاء من عمل هذه العملية فإن الإجراء يقوم بعرض الرسالة التي توضح أن أحد الأمثلة الخاصة بوورد قد تم تشغيلها. إلا أن وورد لا يظهر على Taskbar إلا عند تعيين خاصية Visible على True. وبعد إغلاق مربع الرسالة فإن الإجراء يقوم بإظهار الإطار مع استخدام طريقة Open لمتغير كائن Document لفتح أحد المستندات. كما أن الإجراء يقوم بتعزيز الارتباط بين متغير الكائن ومثال وورد عن طريق تعيين متغير الكائن على Nothing. ولا يؤدي تعيين متغير الكائن على Nothing إلى إغلاق وورد حيث أنه عند انتهاء الإجراء يستمر تشغيل وورد.

ولاختبار الإجراء تتبع الخطوات التالية:

١- افتح وحدة نمطية جديدة في أكسس مع تسميتها basAutomation مع إدخال الإجراء LaunchWord الموضح فيما سبق (قد تحتاج إلى سطر strdoc للفهرس الذي قمت بتثبيت وورد فيه).

٢- اختر Tools ⇦ Refrences ثم انقر مربع الخيار Microsoft Word 9.0 Object Library. إذا كان هذا ضرورياً لتعيين المرجع إلى وورد. انقر OK لإغلاق الحوار.

٣- اضغط على Ctrl+G لعرض الإطار الحالي ثم اكتب LaunchWord مع الضغط على Enter.

ملاحظة

يسلك خادم COM مسلك مختلف حيث أن بعضها تظهر في قائمة المهام حتى إذا كانت قد تم تشغيلها في إطار مخفي. والبعض الآخر مثل وورد وإكسل لا تظهر في لا تظهر في قائمة المهام حتى تقوم بإظهارها.

٤- انقر زر وورد من Taskbar فيتم استعادة إطار Word مع عرض إطار مستند Document.

٥- قلل حجم وورد. حيث سوف يتم استخدام المثال الخاص بوورد فيما بعد.

استخدام وظيفة GetObject

تتميز هذه الوظيفة بأنها متعددة الجوانب بشكل أكبر من وظيفة CreateObject حيث أنه عن طريقها يمكنك مضاعفة الأثر على وظيفة CreateObject. عين أحد متغيرات الكائن على أحد الأمثلة الخاصة بكائن خادم COM، أو افتح الكائن الذي تم حفظه في الملف.

والصيغة التالية هي الصيغة الخاصة بالوظيفة GetObject:

```
Set objvar = GetObject(documentname,appname.objecttype)
```

حيث *objvar* هي متغير الكائن، و*docmentname* هي السلسلة التي تحدد المسار الخاص بالملف الذي يحتوي على الكائن المراد استعادته، و*appname.objecttype* هي السلسلة التي تمثل التطبيق وأحد أنواع الكائنات الخاصة التي يستطيع نظام التشغيل التعرف عليها. يجب تضمين واحدة على الأقل من الوسائط. كما أنه من الممكن إنشاء أحد الأمثلة الجديدة عن طريق الإشارة إلى أحد الأمثلة التي تم تشغيلها من خادم COM، أو افتح أحد الملفات المحددة.

إنشاء مثال جديد إذا قمت بتعيين اسم المستند على سلسلة ليس لها طول فإن وظيفة GetObject تقوم بمضاعفة مسلك الوظيفة CreateObject عن طريق إنشاء مثال آخر جديد لنوع الكائن المحدد. وإذا قمت بتحديد كائن Application فإن وظيفة GetObject تقوم بإنشاء أحد الأمثلة على تطبيق الخادم. وإذا قمت بتعيين أحد الكائنات الأخرى التي يستطيع نظام التشغيل التعرف عليها فإن وظيفة GetObject تقوم بإنشاء مثال مخفي للخادم. ثم يقوم بإنشاء أحد الأمثلة للكائن المحدد مع إعادة المرجع إلى الكائن المحدد.

ومن أحد الأمثلة على ذلك الإجراء GetAccess الموضح فيما يلي حيث يقوم ببدء مثال حديث لأكسس:

```
Public Sub GetAccess()
    Dim appAccess As Access.Application
    Set appAccess = GetObject("", "Access.Application")
    MsgBox "Access is running"
    appAccess.Visible = True
    appAccess.NewCurrentDatabase "NewDatabase"
End Sub
```

بعد إغلاق مربع الرسالة فإن الإجراء يقوم بإظهار إطار أكسس مع استخدام طريقة NewCurrentDatabase لإنشاء قاعدة بيانات جديدة اسمها NewDatabase. ويومض إطار Database لقاعدة البيانات الجديدة لمدة ثانية. وعند انتهاء الإجراء فإن المثال الخاص بأكسس ينتهي. وإذا كانت هناك حاجة إلى ترك المثال يعمل في الذاكرة عند إنهاء الإجراء قم بالإعلان

عن متغير الكائن appAccess في الجزئية Declaration من الوحدة النمطية بدلاً من الإعلان عن المتغير في الإجراء.

ولاختبار الإجراء تتبع الخطوات التالية:

١- ادرج الإجراء GetAccess في الوحدة النمطية basAutomation.

٢- اكتب GetAccess في الإطار الحالي مع الضغط على Enter.

٣- قم بتحديد موضع ملف NewDatabase ثم احذفه. يقوم الإجراء بحفظ ملف NewDatabase في الملف Default Database Folder في علامة التبويب General من الحوار Options "يتم عرضه عن طريق اختيار Tools ⇦ Options". يمكنك استخدام حوار Options لتغيير المجلد الافتراضي لقاعدة البيانات الجديدة.

٤- قم بتعديل الإجراء GetAccess عن طريق قطع عبارة الإعلان عن المتغير ولصقها في الجزء Declaration من الوحدة النمطية. قم بتشغيل الإجراء GetAccess في الإطار الحالي. وعند انتهاء الإجراء يتم عرض قاعدة البيانات في إطار أكسس.

٥- قم بإغلاق المثال في الثاني في أكسس.

الإشارة إلى المثال الذي تم تشغيله لخادم Automation إذا قمت بحذف الوسيطة *documentname* في الوظيفة GetObject فإنها تقوم بإعادة المرجع إلى المثال الخاص بالكائن الذي تم تحديده كما أنه لا يبدأ مثال خادم COM. وإذا لم يكن الخادم في حالة العمل فإن وظيفة GetObject تقوم بإظهار رسالة وقت التشغيل. فمثلاً العبارة التالية تقوم بإعادة المرجع إلى التطبيق Visio عند تشغيل الإجراء، وإلا فإنه فإنها تقوم بإظهار رسالة الخطأ.

```
Dim appVis As Visio.Application
Set appVis = GetObject(,"Visio.Application")
```

ملاحظة عند استخدام الوظيفة GetObject للوصول إلى تطبيقات خادم COM الذي قد تم تشغيله بالفعل، فإنك تقوم بحذف الوسيطة الأولى من وظيفة GetObject تماماً إلا أنه يجب تضمين الفاصلة كعنصر نائب.

ومن أحد الأمثلة على ذلك الإجراء RunningWord الموضح فيما يلي حيث يقوم بالإشارة إلى المثال المستعاد من وورد مع فتح المستند التالي:

```
Public Sub RunningWord()
   Dim appWord As Word.Application, strdoc As String
```

```
    Set appWord = GetObject(, "Word.Application")
    strdoc = "c:\Program Files\Microsoft Office\Office\"
    strdoc = strdoc & "Samples\Formaggi.htm"
    appWord.Documents.Open (strdoc)
    Set appWord = Nothing
End Sub
```

ولاختبار هذا البرنامج، قم بإدراجه في الوحدة النمطية Automation ثم قم بتشغيلها في الإطار الحالي. وعندما ينتهي البرنامج قم باستعادة إطار وورد. وسوف ترى أن هذا الإطار يقوم بعرض المستند Formaggi HTML. قم بإغلاق المسند Formaggi.

فتح ملف محدد إذا قمت بتحديد اسم المسند في وظيفة GetObject فإنها تقوم بإنشاء مثال مخفي جديد لخادم COM مبني على ملحق ملف المستند. وتقوم الوظيفة بفتح الملف واستعادة الكائن إلى الملف. فمثلاً تقوم العبارة التالية بفتح أحد الأمثلة في إكسل وفتح الملف Updates.xls وإعادة المرجع إلى كائن Workbook.

```
Dim objvar As Workbook
Set objvar = GetObject("c:\Excel\updates.xls")
```

إذا كان التطبيق قد تم تشغيله بالفعل فإن وظيفة GetObject تبدأ أحد الأمثلة الأخرى. وعندما يكون للملف أكثر من نوع للكائن يستطيع نظام التشغيل التعرف عليها فإنه من الممكن تحديد أي الكائنات التي تريد إنشاؤه عن طريق استخدام الوسائط أو عن طريق الإعلان عن متغير الكائن بنوع الكائن المحدد. فمثلاً يمكن أن يحتوي ملف إكسل على كل من أنواع الكائن Chart وWorkSheet. كما أن العبارات التالية تقوم أيضاً بإنشاء أمثلة إكسل افتح المصنف Updates.xls مع إعادة المرجع إلى المصنف:

```
Dim wksXL As Object
Set wksXL = GetObject("c:\Excel\updates.xls","Excel.Workbook")
```

تلميح تتم عملية إعادة المرجع إلى كائن أقل في الترتيب الهرمي بسرعة أكبر من إعادة المرجع إلى الكائن الأعلى في الترتيب الهرمي.

استخدام خاصية الأصل: عند فتح أحد الكائنات التي يأتي ترتيبها في موقع سفلي من الترتيب الهرمي للتطبيق فإنه يمكن اجتياز الهيكل الهرمي لنموذج كائن الخادم في كل من الاتجاهين. وللاجتياز في الاتجاه العلوي من الهيكل استخدم خاصية Parent. فمثلاً إذا كنت تعامل مع ورقة نمط معينة فإنه من الممكن أن الوصول إلى المصنف الذي يحتوي على ورقة النمط عن طريق

استخدام خاصية Parent وللوصول إلى ورقة أخرى أو أحد المخططات في نفس ورقة العمل فإنه يمكنك استخدام خاصية Parent للاجتياز إلى أعلى في الهيكل الهرمي في المصنف ثم الإشارة إلى أحد الكائنات الأخرى في ورقة العمل.

من أحد الأمثلة على ذلك فإن إجراء GetSpreadsheet الموضح فيما يلي يستخدم وظيفة GetObject لفتح جدول بيانات في إكسل.

```
Public Sub GetSpreadsheet()
    Dim wks As Workbook, strwks As String
    strwks = "c:\Program Files\Microsoft Office\Office\"
    strwks = strwks & "Examples\samples.xls"
    Set wks = GetObject(strwks)
    wks.Parent.Visible = True
    wks.Windows(1).Visible = True
    wks.Worksheets("Worksheet Functions").Select
    Set wks = Nothing
End Sub
```

يستخدم هذا الإجراء وظيفة GetObject لفتح أحد الأمثلة المخفية في إكسل. انقر Yes في رسائل إكسل للسماح لرسائله بتشغيل الماكرو في المصنف. يقوم إكسل بفتح المثال المخفي للمصنف Sample.xls "تستطيع الوظيفة GetObject فتح إكسل لأن ملحق الملف هو xsl.". يستخدم الإجراء خاصية Parent لاجتياز هيكل الكائن في الاتجاه إلى أعلى للإشارة إلى إكسل مع إظهار إطار التطبيق. وعلى الرغم من أن مثال إكسل الآن مرئياً إلا أن الإطار الذي يحتوي على الملف ليس مرئياً. ولجعل ورقة العمل مرئية فإنه يجب أولاً إظهار المصنف. وفي إكسل ليس هناك خاصية Visible، إلا أنه من الممكن استخدام طريقة Windows للإشارة إلى من كائن Workbook للإشارة إلى النافذة التي تحتوي على المصنف. ثم استخدم خاصية Visible من كائن Window لإظهار الإطار. ويشير الإجراء إلى ورقة العمل Worksheet Functions بعد جعل إطار ورقة العمل مرئياً مع استخدام طريقة Select لجعلها ورقة النمط النشطة.

لاختبار إجراء GetSpreadsheet ادخل الوحدة النمطية Automation مع تشغيلها في الإطار الحالي. وعند الانتهاء من ذلك قم بتصغير المثال الذي قد قمت بتشغيله في إكسل.

تحديد ما إذا كان أحد التطبيقات يعمل من عدمه يمكن استخدام وظيفة GetObject لتحديد ما إذا كان أحد الأمثلة الخاصة بأحد التطبيقات يعمل بالفعل مع التأكد من أنه هناك مثال واحد فقط هو الذي يعمل. من أحد الأمثلة على ذلك الإجراء TestInstance الموضح فيما يلي، حيث يقوم بتشغيل معالج الأخطاء ثم استخدام وظيفة GetObject التي تقوم بحذف اسم المستند لبدء إكسل.

```
Public Sub TestInstance()
    Dim appXL As Excel.Application
    On Error Resume Next
    ' If the application is not running, an error occurs
    Set appXL = GetObject(, "Excel.Application")
    If Err.Number<>0 Then
        Set appXL = CreateObject("Excel.Application")
        appExcel.Visible = True
        MsgBox "Excel is running"
    End If
End Sub
```

وإذا لم يتم تشغيل إكسل فإنه يتم عمل خطأ وقت التشغيل كما أن الإجراء يستخدم وظيفة CreateObject لفتح إكسل وإظهار إطاره. أما إذا كان إكسل قد تم تشغيله فإن الإجراء ينتهي فقط.

ولاختبار الإجراء TestInstance ادخل الوحدة النمطية basAutomation مع تشغيلها في الإطار الحالي، فيقوم الإجراء بتتبع المثال الذي تم تشغيله في إكسل من غير أن تقع أية أخطاء وينتهي الإجراء. ثم اغلق المثال الذي تم تشغيله وقم بتشغيل الإجراء TestInstance. فتقوم في هذه المرة الوظيفة GetObject بتوليد أحد الأخطاء لأن تطبيق إكسل لم يتم تشغيله. ويقوم الإجراء بإنشاء مثال جديد من إكسل مع إظهاره. ويتم إخفاء حوار الرسالة خلف الإطار الذي يعرض مثال إكسل الجديد. عندما تقوم بإغلاق حوار الرسالة فإن ذلك يؤدي إلى إنهاء البرنامج كما يؤدي إلى إنهاء مثال إكسل.

استخدام الكلمات الأساسية الجديدة لإنشاء كائن مضمن يمكن أيضاً إنشاء أحد الأمثلة الجديدة من كائن ActiveX عن طريق استخدام الكلمة الأساسية New عند الإعلان عن متغير الكائن، كما أنها تقوم بإنشاء مثال لكائن ActiveX مع تعيين أحد المراجع إلى متغير الكائن الذي تقوم بالإعلان عنه. كما يمكن أيضاً استخدام New للتطبيقات التي تدعمها فقط مثل أكسس وفيجوال بيسيك. يبدأ VBA أحد الأمثلة الجديدة من التطبيق في كل مرة تستخدم فيها New "إلا إذا كان التطبيق يعتمد على تطبيق واحد فقط".

فمثلاً يفتح الإجراء NewAccess الموضح فيما يلي أحد الأمثلة الأخرى المخفية من أكسس مع إظهار الإطار.

```
Public Sub NewAccess()
    Dim applAccess As New Access.Application
    Dim strfile As String
```

```
    applAccess.Visible = True
    strfile = "c:\Program Files\Microsoft Office\Office\Samples\"
    strfile = strfile & "Northwind.mdb"
    applAccess.OpenCurrentDatabase strfile
    MsgBox "Access is running"
End Sub
```

ملاحظة يفترض مسار الإجراء NewAccess أن أوفيس ٢٠٠٠ قد تم تثبيته. وإذا قمت بتثبيت أكسس ٢٠٠٠ في أحد المجلدات الأخرى فإنك سوف تحتاج إلى تغيير المسار.

ادخل الإجراء NewAccess في الوحدة النمطية basAutomation ثم قم بتشغيلها في الإطار الحالي. يفتح إطار أكسس مع عرض قاعدة البيانات Northwind في إطار أكسس، كما أنه يظل مفتوحاً في أثناء تشغيل الإجراء. ويكون حوار الرسالة في أسفل الإطار الذي يعرض المثال الجديد من أكسس. وعند إغلاق مربع الحوار ينتهي التطبيق. وعند انتهاء الإجراء فإنه يتم إلغاء متغير الكائن appAccess ومثال أكسس الذي قام الإجراء بعمله.

إصدار الكائن

عندما يستخدم الإجراء VBA كائنات ActiveX من تطبيقات خادم COM فإن التطبيق يصبح مسؤولاً عن فتح وإغلاق الكائن. وعند الانتهاء من الكائن فإنه من المستحسن إغلاق الكائن والخروج من التطبيق.

يمكن تقوية الارتباط بين الكائن والمتغير عن طريق تعيين متغير الكائن على Nothing باستخدام العبارة:

```
Set objvar = Nothing
```

إلا أنه عند تشغيل VBA لهذه العبارة فإن الرابطة يتم تقويتها، كما أن الكائن نفسه ربما ينتهي. وتتصرف الكائنات على نحو مختلف في هذا الخصوص. فعلى سبيل المثال فإن كل من إكسل وباور بوينت وورد تستمر في العمل بعد تعيين متغير الكائن على Nothing أو بعد إنهاؤه عند انتهاء التطبيق. وبالعكس فإنه عند إنهاء أحد متغيرات الكائن في أكسس فإن ذلك يؤدي إلى إنهاء التطبيق الذي تم بدؤه في باستخدام Automation.

يكون لكل تطبيق طريقة وهي دائماً Close أو Exit تمكن من الخروج منه. وتحتوي أغلب كائنات ActiveX على طريقة Close التي تتيح إغلاق الكائن. فمثلاً إذا كنت تستخدم عند التعامل مع إكسل فإنه يجب أولاً استخدام طريقة Quit لإغلاق التطبيق ثم تعيين متغير الكائن لإصدار المرجع على النحو التالي:

```
appExcel.Quit
Set appExcel = Nothing
```

تم في هذا الجزء توضيح التقنيات الأساسية للتعامل مع تطبيقات خادم COM، كما أنه قد تم التعرض لكيفية فتحها وإغلاقها وكيفية إنشاء كائنات ActiveX وإنهائها. ويجب معرفة أن كتابة التعليمات للتعامل مع كائنات ActiveX لتطبيق آخر يحتاج إلى معرفة جيدة بنوع كائن التطبيق بما في ذلك الخصائص والطرق والأحداث الخاصة بهذا الكائن.

خلاصة

يوضح هذا الفصل بعضاً من الموضوعات المتقدمة التي تتطلب استخدام أدوات VBA البرمجية. ويبدأ هذا الفصل بمناقشة كيفية تحويل الماكرو إلى الإجراء مما يتيح لك استخدام المميزات التي تخص VBA مع تعزيز الإمكانات الخاصة بإنشاء الماكرو. ثم يقدم التقنيات الخاصة بزيادة وظائف أكسس عن طريق تضمين مميزات من مصادر أخرى مثل مكتبة قواعد البيانات ومكتبات التعليمات البرمجية الأخرى وعناصر تحكم ActiveX وكائنات ActiveX الموجودة في الكائنات الأخرى "مكونات ActiveX". ومن النقاط الهامة في هذا الفصل:

- يمكنك استخدام الوظائف والإجراءات المخزنة في DLL.
- إذا كان DLL يحتوي على مكتبة النوع فإنه من الممكن أن يتم تعيين أحد المراجع إليها مع استخدام الإجراءات بنفس الطريقة التي تستخدم بها الإجراء في التطبيق.
- إذا لم يكن هناك مكتبة للنوع فإنه يجب الإعلان عن الإجراء في الوحدة النمطية التي تلي الصيغة المحددة التي تتطلبها المكتبة.
- يمكن إضافة بعد الإضافات إلى المشروع عن طريق إدراج عناصر تحكم ActiveX في النموذج، حيث أن لها خصائصها وطرقها وأحداثها الخاصة.
- يمكن استخدام VBA لأكسس للتحكم في أحد التطبيقات الأخرى. مثل إكسل أو وورد عن طريق استخدام Automation.

ومع مزيد من الممارسة والتطبيق يمكنك استخدام كل التقنيات التي تم عرضها في هذا الكتاب لإنشاء تطبيقات مخصصة غير بسيطة. وبالإضافة إلى ما يعرضه هذا الكتاب من أساسيات استخدام عناصر برمجة VBA فإنه أيضاً يعرض مقدمة مختصرة عن بعض الموضوعات والتقنيات التي تتميز بكونها أكثر تعقيداً فيما يخص هذا الشأن.

ملحق أ

مسرد

A

تجريد "abstraction"

تكوين نموذج لكائن يشتمل على الخصائص والوسائل المرتبطة بغرض فيه أو بوظيفته متجاهلاً النواحي الأخرى.

وسيط إجراء "action argument"

معلومة إضافية يتم اختيارها ككائن أو معرف أو تعبير حرفي يستخدم لإجراء ماكرو. ولا تشتمل وسائط الإجراء على أي متغيرات.

ActiveX

مجموعة من تقنيات مايكروسوفت للاتصال بين مكونات البرامج.

مكون ActiveX

تطبيق مثل مايكروسوفت أكسس وإكسل وFrontPage يوفر كائنات برمجية لتطبيقات أخرى أو يستخدم الكائنات البرمجية التي تتيحها التطبيقات الأخرى.

عنصر تحكم ActiveX

عنصر تحكم يمكن إضافته لنموذج معين لإتاحة إمكانيات إضافية لا يوفرها أكسس. و يتمتع عنصر تحكم ActiveX بخصائصه وطرقه وأحداثه الخاصة. ويمكن كتابة إجراءات VBA لتناول عناصر تحكم ActiveX.

كائن ActiveX

كائن برمجي يورده تطبيق خارجي ويمكن التحكم فيه بواسطة التطبيق الحالي باستخدام الأتمتة.

عنوان "address"

عدد أو نقش من بت يضع تعريفاً فريداً لموقع في ذاكرة الكمبيوتر. أنظر أيضا: عنوان ارتباط تشعبي.

مجموعة أحرف ANSI

مجموعة أحرف من ٨ بت تستخدم في ويندوز لتمثيل أحرف على لوحة المفاتيح يصل عددها لـ ٢٥٦. قام بتطويرها معهد المعايير الوطنية الأمريكية "ANSI".

وسيطة "argument"

ثابت أو تعبير يوفر معلومات لأمر أو إجراء أو إجراء ماكرو أو إجراء أو وسيلة. وقد تشتمل وسائط الإجراءات والوسائل على متغيرات. وعادة ما يطلق اسم "معاملات" على الوسائط المرسلة لإجراءات VBA.

صفيف "array"

سلسلة من المتغيرات المفهرسة تستبدل واحدة تلو الأخرى بنفس اسم المتغير ويتم معالجتها بالتتابع حسب ما يحدده استخدام الإجراء لاسم هذا المتغير.

استعلام بحث تلقائي "AutoLookup query"

استعلام قائم على جدول أو أكثر في علاقة رأس بأطراف التي تقوم تلقائيا بملأ قيم الحقول في الجدول الواحد عند إدخال قيمة في حقل مرتبط لسجل جديد.

الأتمتة "Automation"

مجموعة من قواعد وأساليب برمجية للتحكم في كائنات تطبيق آخر.

B

جدول أساسي "base table"

جدول في قاعدة بيانات مايكروسوفت جيت "ملف بامتداد mdb." كما يسمى جدول محلي.

إشارة مرجعية "bookmark"

أي وسيلة يمكن استخدامها لحفظ المكان الحالي. في الوحدة النمطية الوسيلة المستخدمة للقيام بهذه المهمة هي عرض رمز في الهامش الأيسر. أما في كائن نموذج أو مجموعة سجلات فإن تخزين سلسلة ثنائية هو الوسيلة التي تعرف السجل الحالي. وتتاح خاصية الإشارة المرجعية للنموذج في VBA.

عنصر تحكم منضم "bound control"

عنصر تحكم في نموذج أو تقرير يحصل على بياناته من حقل أو عمود في الجدول أو أسلوب العرض أو الاستعلام الأساسي أو جملة SQL خاصة بنموذج أو بجزء صفحة الوصول للبيانات أو بتقرير. ويتمثل إعداد خاصية ControlSource لعنصر تحكم منضم في اسم الحقل أو العمود.

وضع التوقف "break mode"

حالة يتم فيها تشغيل برنامج VBA لكن يتم تعليقه أثناء تنفيذ العبارات.

نقطة الإيقاف "breakpoint"

سطر برنامج في إجراء يتم عنده التعليق التلقائي للتنفيذ قبل تشغيل العبارة. يمكن إعداد نقطة إيقاف لاستكشاف وإصلاح الأخطاء الموجودة في برنامج.

حسب المرجع "by reference"

أحد وسائل تمرير عنوان وسيطة إلى إجراء، ويمكن للإجراء الوصول إلى المتغير الفعلي وتغيير قيمته. وحسب المرجع هو الطريقة الافتراضية لتمرير وسيطة معينة.

حسب القيمة "by value"

أحد طرق تمرير قيمة أو محتوى وسيطة بدلا من عنوانها إلى إجراء. وبهذه الطريقة يصل الإجراء إلى نسخة من المتغير لكنه لا يقوم بتغيير قيمته المخزنة.

C

ذاكرة تخزين مؤقت "cache"

مساحة من الذاكرة المحلية يوجد عليها أحدث بيانات تم استدعائها من الملقم في حالة طلبها مرة ثانية بينما يكون التطبيق في حالة التشغيل. ويقوم محرك قاعدة البيانات بالبحث عن البيانات المطلوبة في ذاكرة التخزين المؤقت قبل البحث عنها على القرص.

عنصر تحكم محتسب "calculated control"

عنصر تحكم في نموذج أو تقرير يقوم بعرض نتيجة العملية الحسابية بدلا من البيانات المخزنة. وإعداد خاصية ControlSource بالنسبة لعنصر تحكم محتسب هو تعبير العملية الحسابية.

شجرة الاستدعاءات "call tree"

كل الوحدات النمطية التي تحتوي على إجراءات يمكن استدعائها بإجراء في وحدة نمطية جاري تشغيل البرنامج فيها.

تحسس حالة الأحرف "case-sensitive"

القدرة على التفرقة بين الأحرف الكبيرة والصغيرة.

فئة "class"

التعريف الخاص لمجموعة كائنات ويشمل الخصائص والوسائل والأحداث. ويطلق على الكائن القائم على تعريف *مثال* للفئة.

وحدة نمطية للفئات "class module"

وحدة نمطية تقوم بتعريف مجموعة كائنات. يحتوي أكسس 2000 على وحدات نمطية للفئات خاصة بالنماذج والتقارير ومجموعات الكائنات المستقلة. كما تقوم الوحدة النمطية للفئات المستقلة بتحديد مجموعة الكائنات الغير مرتبطة بالنماذج ولا بالتقارير.

اسم الفئة "class name"

الاسم المستخدم للدلالة على وحدات نمطية لفئات معينة. على سبيل المثال الفئة الخاصة بالوحدة النمطية للنماذج التي تسمى اسم النموذج يطلق عليها Form_FormName.

فرع "clone"

مجموعة سجلات لها مؤشر سجل حالي مستقل.

فهرس تفاوت المسافات "clustering index"

فهرس يحتوي على حقل غير أساسي أو أكثر ويقوم عند تجميعه بترتيب كل سجلات الجدول بطريقة معرفة مسبقا. ويوفر فهرس تفاوت المسافات وسيلة فعالة للوصول للسجلات في الجدول خاصة عندما تكون قيم فهرس الجدول غير فريدة.

تعليمات برمجية "code"

مجموعة التعليمات النصية التي يتم إدخالها في الوحدات النمطية للتطبيق وتسمى أيضا *البرنامج المصدر*. وتشتمل التعليمات البرمجية على إعداد خيارات الوحدات النمطية والتعليقات والجمل الخاصة بالثوابت والمتغيرات وإعلانات الإجراءات وبنيات عناصر التحكم والتعيينات وأساليب التنفيذ.

قالب التعليمات البرمجية "code template"

السطران الأول والأخير من الإجراء، ويتم إنشائهما باستخدام منشئ البرامج أو اختيار أمر إجراء من قائمة إدراج.

مجموعة "collection"

كائن يشتمل على مجموعة من الكائنات المتماثلة. على سبيل المثال تشتمل مجموعة النماذج على زمرة من النماذج المفتوحة "زمرة كائنات النماذج".

شريط الأوامر "command bar"

واحد من ثلاثة وسائل لعرض الأوامر المضمنة والمخصصة وتشمل أشرطة القوائم وأشرطة الأدوات والقوائم المنبثقة.

وقت التجميع "compile time"

الوقت الذي يتم فيه ترجمة تعليمات VBA البرمجية إلى برنامج تشغيل. ويختلف وقت التجميع حسب صعوبة تعليمات VBA البرمجية والأخطاء التي قد تحدث أثناء بناء الجمل. وقد يكون هناك حاجة لإصلاح هذه الأخطاء وإعادة تجميع تعليمات VBA البرمجية لاختبار التغييرات التي تم إجرائها.

وضع مجمع "compiled state"

إصدار من التعليمات البرمجية يتم إنشائه بواسطة برنامج تجميع VBA. يقوم برنامج التجميع بترجمة النص الموجود في الوحدات النمطية "البرنامج المصدر" إلى لغة أقرب إلى لغة الآلة ويجعلها أكثر كفاءة من البرنامج المصدر. "يقوم أكسس بتشغيل الوضع المجمع وليس البرنامج المصدر."

موجه برنامج التجميع "compiler directive"

عبارة في التعليمات البرمجية مسبوقة برمز رقم يمكن استخدامها لاستبعاد عبارة من عملية التجميع. وهذا يفيد في حالة حذف العبارات الخطأ واختبار الإصدارات المصححة كل على حدة.

سلسلة "concatenation"

عملية ربط السلاسل. فمثلاً لتكوين اسم بالكامل من الاسم الأول والاسم الأخير يتم سلسلة "جـون" و"نوفاليز" للحصول على "جون نوفاليز".

سلسلة الاتصال "connection string"

سلسلة تحتوي على معلومات ضرورية لفتح قاعدة بيانات خارجية مثل المسار الموصل للملف.

عنصر تحكم "control"

كائن رسومي يوضع في نموذج أو صفحة قاعدة بيانات أو تقرير لعـرض البيانـات أو لـتزيين النموذج أو الصفحة أو التقرير أو للقيام بإجراء معين.

بنية عنصر التحكم "control structure"

العبارات التي يتطلبها تغيير الترتيب الخاص بتنفيذ عبارات التعليمات البرمجية.

مخزن النسخ المؤقت "copy buffer"

موقع في الذاكرة ينشئه محرك قاعدة البيانات يشتمل على محتويات السجل الجـاري تحريـره. وتقوم وسيلة Edit بنسخ السجل الحالي لمخزن النسخ المؤقت، كما تقوم وسيلة AddNew بمسـح المخزن لاستقبال السجل الجديد وإعداد القيم الافتراضية، وتقوم وسيلة Update بحفظ البيانـات في مخزن النسخ المؤقت وذلك باستبدال السجل الحالي أو بإدراج السجل الجديد. وأي جملة تعيـد إعداد أو نقل مؤشر السجل الحالي تتجاهل محتويات مخزن النسخ المؤقت.

الفهرس الحالي "current index"

أحدث فهرس يتم إعداده باستخدام خاصية فهرس "لكائن مجموعة سجلات مفهرسـة مـن نـوع جدول". يمكن أن يكون لكائن مجموعة السجلات عدة فهارس لكنه لا يستخدم أكثر من واحد فـي نفس الوقت.

السجل الحالي "current record"

سجل في مجموعة سجلات يمكن تعديله واستدعاء معلومات منه. سجل واحد فقط في مجموعة السجلات يمكن أن يكون السجل الحالي، إلا أنه لا يشترط أن يكون لمجموعة السجلات سجلا حاليا. استخدم أساليب ... Move لتغيير وضع السجل الحالي في مجموعة السجلات. استخدم أساليب ... Find مع مجموعة سجلات حيوية أو صور ثابتة أو أسلوب Seek مع مجموعات سجلات من نوع جداول لتغيير وضع السجل الحالي عند الحاجة لذلك.

D

مصدر البيانات "data source"

مصدر البيانات الخاصة بعنصر تحكم أو نموذج أو تقرير أو جزء صفحة الوصول لقاعدة بيانات أو قاعدة بيانات.

نوع البيانات "data type"

السمة المعرفة للمتغير ووظيفتها تحديد نوع البيانات التي يمكن أن يشملها المتغير.

تصحيح "debugging"

عملية تحديد مكان مصدر الخطأ في إجراء وإزالته.

إعلانات "declaration"

عبارات تحدد الثوابت والمتغيرات وأنواع البيانات المعرفة من قبل المستخدم والإجراءات الخارجية من مكتبة الربط الحيوي ".DLL" المتاحة لإجراءات الوحدة النمطية. وتوضع هذه العبارات في الجزء الخاص بالإعلانات في الوحدة النمطية.

مقطع الإعلانات "Declaration section"

جزء من وحدة نمطية يشتمل على إعلانات تنطبق على كل الإجراءات في الوحدة.

قيمة افتراضية "default value"

القيمة التي يتم إدخالها تلقائيا في حقل أو عنصر تحكم عند إضافة سجل جديد.

المحددات "delimiters"

الرموز المستخدمة لتحديد بداية ونهاية مقطع معين أو عناصر مجمعة في برنامج مثل علامات الترقيم التي تطوق سلسلة النص.

المجال "domain"

مجموعة من السجلات في مصدر بيانات ترتبط ببعضها وتعالج كمجموعة بطريقة معينة .

دالة تجميع المجال "domain aggregate function"

دالة تقوم بحساب إحصائية معينة مثل الجمع أو تحديد الأكبر قيمة على أساس القيم في مجموعة السجلات المحددة التي تسمى *مجال*.

مكتبة الربط الحيوي "dynamic-link library "DLL""

ملف خارجي يشتمل على إجراءات يمكن للتطبيق الاتصال بها واستخدامها وقت التشغيل.

مجموعة سجلات حيوية "dynaset-type recordset"

مجموعة حيوية من السجلات تمثل جدول في قاعدة البيانات المفتوحة أو في الجدول المرفق أو نتيجة تشغيل استعلام أو جملة SQL SELECT. وعند استخدام كائن مجموعة سجلات حيوية فإن الكائن في هذه الحالة لا يحتوي إلا على مجموعة من المراجع أو القيم الأساسية، ولا يتم استدعاء السجل بأكمله إلا عند الحاجة لتحريره أو عرضه.

E

الصدى "echo"

عملية تحديث الشاشة أو تقديم معلومات للمستخدم أثناء تشغيل الإجراء.

فارغ "empty"

حالة متغير مهيأ من نوع بيانات الأشكال المختلفة. عند بداية الإجراء يقوم أكسس بحجز مساحة لكل المتغيرات التي تم إعلانها صراحة وبإعدادها على فارغ.

تغليف "encapsulation"

تغليف الخصائص والوسائل كعناصر داخلية للكائن. ويعد التغليف أحد العناصر الأساسية للبرمجة الموجهة للكائنات.

خطأ "error"

انحراف عما هو صحيح. ومن الأخطاء التي يمكن تفاديها تلك التي تقع وقت التجميع والتي تقع وقت التشغيل. ومن الأخطاء التي لا يمكن تفاديها تلك التي تحدث بعد إزالة كل الأخطاء، وتقع مثل هذه الأخطاء بسبب المستخدم أو اختلال الطاقة أو مشاكل في الشبكة.

تعليمات برمجية للأخطاء "error code"

قيمة مكونة من عدد صحيح تقوم بتعريف الخطأ الذي يحدث وقت التشغيل تعريفاً فريداً.

معالج الأخطاء "error handler"

مقطع من التعليمات البرمجية مميز بعنوان السطر أو رقم السطر يحدد الإجراءات التي ستتبع في حالة حدوث أخطاء.

معلومات برمجية لمعالجة الأخطاء "error-handling code"

جمل VBA تحدد التعليمات البرمجية الخاصة بالخطأ ثم تقوم بالمقاطعة وتعديل الاستجابة الافتراضية لخطأ معين.

حدث "event"

تغيير في حالة الكائن ناتجة عن بعض الإجراءات التي يتيحها أكسس كإمكانيات برمجية مثل حدث Click وذلك عندما يتم تعريفه بواسطة أحد أزرار الأوامر. ويمكن تعيين إجراء أحداث أو إجراء دالة أحداث لتنفيذها عندما يتعرف الكائن على التغيير في حالته.

إجراء أحداث "event procedure"

نوع من الإجراءات الفرعية يتم تنفيذها تلقائيا استجابة لحدث يطلقه المستخدم أو البرنامج المصدر أو النظام. ولإجراءات الأحداث أسماء وقوائم وسائط معرفة مقدما.

خاصية الحدث "event property"

سمة ذات أسم تتعلق بعنصر تحكم أو نموذج أو مقطع مرتبط بحدث يمكن للبند أن يستجيب لها. ويمكن تشغيل إجراء عندما يتم تعريف الحدث بإعداد خاصية الحدث المرتبط على [Event Procedure] أو الدالة [=funtionname()].

برنامج تشغيل "executable"

برنامج يستخدمه نظام التشغيل لتنفيذ مجموعة معينة من التعليمات. ولا تعد التعليمات البرمجية برنامج تشغيل إلا بعد تجميعها.



تصفية "filter"

مجموعة من المعايير تستخدم لتصنيف مجموعة من السجلات أو لتحديد مجموعة فرعية من السجلات. على سبيل المثال يمكن استخدام التصفية لتحديد الطلبات التي تم وضعها بعد تاريخ

معين أو الطلبات التي وضعها عميل معين.

التركيز "focus"

إمكانية أخذ مدخلات من المستخدم من خلال إجراء ماوس أو لوحة مفاتيح. ولا يكون التركيز إلا لبند واحد في نفس الوقت.

مفتاح ربط "foreign key"

حقل أو مجموعة من الحقول في جدول أو استعلام تتوافق قيمهم مع قيم مفتاح أساسي في جدول أو استعلام آخر عند ربط الجدولين أو الاستعلامين.

استعلام أو جدول ربط "foreign table or query"

جدول أو استعلام يشارك في علاقة كجدول أو استعلام "كثير" ويشتمل على الحقل الذي يوافق حقل المفتاح الأساسي في الاستعلام أو الجدول المرتبط.

وحدية نمطية للنماذج "form module"

وحدة نمطية تقوم بتخزين التعليمات البرمجية لكل إجراءات الأحداث التي تطلقها الأحداث الموجودة في نموذج معين أو عناصر التحكم داخله. كما تشتمل الوحدة النمطية للنماذج على إجراءات أخرى متعلقة بالنموذج.

مجموعة سجلات من نوع للأمام فقط "forword-only-type recordset"

نسخة ثابتة من مجموعة سجلات قائمة على جدول أو استعلام أو جملة SQL SELECT. وهذا النوع مماثل لمجموعة السجلات من نوع الصور الثابتة إلا أنه يتيح فقط إمكانية التمرير للأمام فقط من خلال السجلات.

إجراء دوال "function procedure"

إجراء ينفذ عمل معين ويرجع قيمة. ويمكن استخدام القيمة المرتجعة لإجراء الدوال في أحد

التعبيرات.

G

متغير عمومي "global variable"

متغير يتم الإعلان عنه في مقطع الإعلانات داخل وحدة نمطية باستخدام الكلمة الأساسية Public.
ويمكن لكل الإجراءات أن ترى المتغير العمومي في الوحدات النمطية لقاعدة البيانات.

معرف عمومي فريد ""GUID" global unique identifier"

حقل مكون من ١٦ بايت يقوم بتعريف للنسخ المماثلة تعريفاً فريداً، ويطلق عليه معرف النسخة المماثلة في أكسس.

H

عنوان الارتباط التشعبي "hyperlink address"

المسار المحدد لكائن أو مستند أو وجهة أخرى معرفة على أنها URL عندما تكون الوجهة موقع على الإنترنت أو الشبكة المحلية أو على أنها موقع على الشبكة عندما تكون الوجهة ملف على الشبكة المحلية.

نوع بيانات الارتباط التشعبي "Hyperlink data type"

نوع بيانات الحقل الخاص بتخزين الارتباطات التشعبية. ويتيح إمكانية اشتمال أكسس على مصادر بيانات HTML.

كائن ارتباط تشعبي "Hyperlink objects"

كائن يمثل ارتباط تشعبي مرتبط بزر أمر أو عنوان أو أمر قائمة أو عنصر تحكم صورة في نموذج، أو مرتبط بمربع نصي أو بجزء مربع النص الخاص بعنصر التحكم الموجود في مربع تحرير وسرد منضم لحقل له نوع بيانات ارتباط تشعبي.

"Hypertext Markup Language" HTML

نظام لتضمين تعليمات تنسيق تسمى علامات "TAGS" في كل جزء من مستند نص لكي يمكن للمستند أن ينشر على الويب باستخدام تنسيقات العرض التي تم تحديدها.

I

إطار فوري "Immediate window"

الإطار الذي يمكن من خلاله تنفيذ واختبار سطور منفردة من تعليمات VBAبرمجية في الحال.

إنشاء ضمني "implicit creation"

إمكانية استخدام الكلمة الأساسية New في جملة الإعلان الخاصة بمتغير الكائن لتحديد مثال جديد من الكائن في أول مرة تقوم فيها جملة الإجراء بالإشارة إلى الكائن.

فهرس "index"

عند استخدامه مع الجداول يقصد به جدول مرجع مرتبط بجدول قاعدة بيانات يربط قيمة في جدول المرجع مع موقع السجل المقابل في الملف الذي يشتمل على جدول قاعدة البيانات. ومع الجداول يقوم الفهرس بتسريع عملية البحث عن سجلات جداول قاعدة البيانات، وعند استخدامه مع المجموعات يقوم الفهرس بتمييز موضع عدد معين بالنسبة للأعداد الأخرى في المجموعة.

إرث "inheritance"

قدرة الكائنات في الفئة التابعة على استخدام خصائص ووسائل الفئة الأصل تلقائيا.

مثال "instance"

كائن منشئ قائم على تحديدات الفئة. على سبيل المثال عند وضع زر أمر في نموذج فإن الكائن الذي تم وضعه على النموذج ما هو إلا مثال لفئة زر الأمر ممثلة بأيقونة زر الأمر الموجودة في مربع الأدوات.

"intrinsic constant" ثابت جوهري

اسم ذو معنى يعطيه تطبيق ليحل محل عدد لا يتغير.

K

"keyword" كلمة أساسية

كلمة تستخدمها لغة الكمبيوتر كجزء منها، على سبيل المثال: الكلمات IF وThen وElse وChoose وSwitch و Select Case تمثل كلمات VBA أساسية لاتخاذ قرارات.

L

"library database" قاعدة بيانات المكتبات

مجموعة من كائنات قاعدة البيانات والماكرو وإجراءات VBA يمكن استدعائها من مشروع آخر لأكسس.

"lifetime" عمر

الفاصل الزمني بين إنشاء متغير عند تخصيص المتغير لقيمة أو مرجع كائن وإزالته.

"lightweight form or report" تقرير أو نموذج خفيف

نموذج غير مرتبط بوحدة نمطية للنماذج أو تقرير بدون وحدة نمطية للتقارير.

"literal" حرفي

رمز يدل على قيمة معينة لا يمكن تغيير تعريفها، وبمعنى آخر لا يمكن إعادة تعريف الرموز الحرفية.

تاريخ حرفي "literal date"

مجموعة من الحروف تحيط بها علامة #، مثال #٦/٦/٩٩#. استخدم هذا الشكل للتواريخ التي لا تريد تعديلها أو تحديثها.

سلسلة حرفية "literal string"

مجموعة من حرف واحد أو أكثر تحيط بها علامات التنصيص مثل "هذه سلسلة حرفية".

مقفل "locked"

حالة كائن صفحة بيانات أو كائن مجموعة سجلات أو كائن قاعدة بيانات تجعله في وضع قراءة فقط لكل المستخدمين ما عدا الشخص الذي داخل البيانات حاليا.

خطأ منطقي "logic error"

خطأ يقع عندما تقوم الإجراءات بتنفيذ الأعمال بدون أن تأتي بالنتائج المقصودة.

تنقل منطقي "logic navigation"

التنقل بين السجلات حسب البيانات داخل السجل بدلا من التنقل حسب وضعها الحقيقي في مجموعة السجلات.

حلقة "loop"

بنية مميزة بجملة بداية ونهاية تتيح تكرار تنفيذ مجموعة من الجمل التي بين الجملتين.

M

ملف mde. ".mde file"

ملف قاعدة بيانات أكسس بعد إزالة البرنامج المصدر ويشتمل فقط على النسخة المجمعة من التعليمات البرمجية.

شريط القوائم "menu bar"

شريط أفقي تحت شريط العنوان يشتمل على قائمة بأسماء مجموعات الأوامر المتوافرة كأوامر قوائم، وعند النقر على اسم قائمة يتم عرض قائمة منسدلة بأوامر القائمة.

مربع الرسالة "message box"

نموذج منبثق مشروط يعرض رسالة ويحتوي على زر أو أكثر. ويمكن نقر أحد الأزرار "غير زر التعليمات" لغلق النموذج أو الاستمرار.

وسيلة "method"

إجراء مرتبط بنوع كائن معين يحدث تأثيره على الكائن.

وحدة نمطية "module"

مجموعة من الإعلانات والجمل والإجراءات مخزنة معاً كوحدة لها اسم واحد.

متغير على مستوى الوحدة النمطية "module-level variable"

متغير تم الإعلان عنه في مقطع الإعلانات داخل الوحدة المنطقية. ويتيح استخدام الكلمات الرئيسية Private أو Public الوحدة النمطية لكل الإجراءات إما داخل الوحدة النمطية أو داخل كل الوحدات على التوالي.

N

مؤقت "nonpersistent"

كائن لا يوجد إلا في الذاكرة فقط ولا يتم حفظه في ملف قاعدة البيانات. على سبيل المثال عند فتح جدول أو إجراء استعلام يتم تكوين كائن مجموعة سجلات، ثم يزال الكائن بمجرد غلق الجدول أو صفحة بيانات الاستعلام.

قيمة خالية "Null"

قيمة خاصة لنوع بيانات الأشكال المختلفة التي تدل على أن البيانات مفقودة أو غير معرفة أو غير مطابقة.



كائن "object"

عنصر يتم تعريفة بواسطة مجموعة من الخصائص ومجموعة من الإجراءات (تسمى وسائل) يمكن للكائن أن يوظفها على نفسه.

برمجة قائمة على الكائنات "object-centric programming"

أسلوب برمجة تقوم فيه الجمل المفردة بالتعرف على الكائن وعمل إجراءات عليه.

برنامج تشغيل ODBC "data source"

مكتبة ربط حيوي تصل ما بين مصدر بيانات معين وتطبيق آخر.

ODBCDirect

تقنية تسمح بتجاوز محرك قاعدة بيانات جيت والاتصال مباشرة بملقم قاعدة بيانات SQL.

الاتصال بقاعدة بيانات مفتوحة [Open Database Connectivity "ODBC"]

مجموعة بروتوكولات للوصول للمعلومات في ملقم قاعدة بيانات SQL مثل محرك قاعدة بيانات جيت أو ملقم SQL لمايكروسوفت.

P

معامل "parameter"

رمز في جملة إعلان الإجراءات أو في معايير الاستعلامات يستبدل بقيم تعطى عند استدعاء الإجراء أو تشغيل الاستعلام.

استعلام معامل "parameter query"

استعلام يتطلب معلومات إضافية قبل تشغيله.

توزيع "parse"

فصل تعبير إلى أجزاء والتعرف عليها كبنود معروفة ويشمل ذلك علاقاتها حسب ما تحدده مجموعة من القواعد.

ملازم "persistent"

كائن يتم تخزينه في ملف قاعدة البيانات، مثال: تعريفات الجداول والاستعلامات والنماذج والتقارير يتم تخزينها على أنها كائنات ملازمة.

تنقل حقيقي "physical navigation"

التنقل بين السجلات حسب موقعها الحقيقي في مجموعة السجلات مثل الموضع الأول والأخير والتالي والسابق وليس على حسب البيانات داخل السجل.

تعدد الأشكال "polymorphism"

تمكن فئتين أو أكثر من أن يكون لها وسائل بنفس الاسم ونفس الغرض مع تعليمات تنفيذ مختلفة. على سبيل المثال كائنات DoCmd و Form و Control لها وسيلة Requery لكنها تنفذ بطريقة مختلفة مع كل حالة.

شرط مسبق "precondition"

شرط لتحديد ما إذا كان تشغيل إجراء أو مجموعة من الإجراءات قد يسبب خطأ وقت التشغيل.

فهرس رئيسي "primary index"

فهرس يتكون من حقل أو أكثر ويقوم بتعريف السجلات الموجودة في جدول تعريفاً فريداً بترتيب محدد مسبقا. ويشتمل الفهرس الرئيسي على مفتاح للجدول وعادةً على نفس الحقول مثل المفتاح الرئيسي للجدول.

جدول أو استعلام رئيسي "primary table or query"

جدول أو استعلام يشترك في علاقة، جرى العرف على أنه جدول أو استعلام "واحد"، ويحتوي على الحقل الموافق لحقل مفتاح الربط في الجدول أو الاستعلام المرتبط.

إجراء "procedure"

تسلسل من الجمل يتم تنفيذها كوحدة واحدة.

متغير على مستوى الإجراء "procedure-level variable"

متغير يتم الإعلان عنه في إجراء. ويتاح المتغير على مستوى الإجراء في الإجراء الذي تم إعلانه فيه فقط وذلك بخلاف المتغيرات على مستوى الوحدات النمطية.

معالجة "process"

مجموعة من التعليمات ينفذها الكمبيوتر في نظام تشغيل متعدد المهام.

مشروع "project"

مجموعة تتكون من كل الوحدات النمطية الخاصة بالتعليمات البرمجية المجمعة. وبشكل افتراضي يتخذ المشروع نفس اسم قاعدة البيانات. وفي أكسس ٢٠٠٠ يشير المشروع أيضا إلى قاعدة بيانات باستخدام النهاية الخلفية لملقم SQL.

اسم المشروع "project name"

الاسم الذي يمكن استخدامه لتعريف كل التعليمات البرمجية المجمعة الموجودة في قاعدة بيانـــات. ويقوم أكسس بشكل افتراضي باستخدام اسم قاعدة البيانات كاسم للمشروع مع إمكانية تحديد اســـم آخر.

إجراءات الخصائص "property procedure"

مجموعة من الإجراءات تنشأ خصائص جديدة للنموذج أو التقرير، فإجراء Property Let يقوم بإنشاء خاصية مخصصة لنموذج أو تقرير. أما إجراء Property Get فيقـــوم بإرجـــاع قيمـــة لخاصية مخصصة لنموذج أو تقرير.

تعليمات برمجية صورية "pseudo-code"

ناتج المعالجة التي يقوم بها مجمع برامج VBA للتعليمات النصية والجمل الموجودة فـــي وحـــدة نمطية.

R

مجموعة سجلات "recordset"

مجموعة من السجلات يرجعها إلى الذاكرة جدول أو استعلام أو عرض أو جملة SQL لها إمكانية إرجاع سجلات.

تحديث "refresh"

إعادة عرض سجلات في صفحة بيانات أو نموذج أو تقرير لعكس التغييرات التي تم إجرائها فـــي إطار آخر أو من قبل مستخدم آخر عند العمل في بيئة متعددة المستخدمين. وتحديث البيانـــات لا يعرض السجلات الجديدة و لا يعيد تشغيل الاستعلام أو الجدول الأساسي.

إعادة استعلام "requery"

إعادة تشغيل الاستعلام أو الجدول الأساسي لصفحة البيانات أو النموذج أو التقرير. وتعكس إعادة

الاستعلام أي تغيير يطرأ على البيانات وتعرض السجلات الجديدة وتزيل السجلات المحذوفة أو السجلات التي لم تعد مناسبة لمعايير الاستعلام.

وضع التشغيل "run mode"

يعرض أكسس ثلاثة أنماط عرض في هذه الوحدة النمطية: صفحة البيانات والنموذج ومعاينة قبل الطباعة.

وقت التشغيل "run time"

الوقت الذي يجري فيه تشغيل إجراء VBA.

خطأ وقت التشغيل "run-time error"

خطأ يحدث عندما يحاول إجراء معين تنفيذ عمل ما لكنه يفشل فيه.

S

مجال "scope"

إمكانية رؤية ثابت أو متغير أو إجراء أو كائن. على سبيل المثال يمكن لإجراء معين أن يرى متغيراً تم الإعلان عنه فيه ولهذا فإن مجاله يتمثل في هذا الإجراء.

برمجة قائمة على اختيارات "selection-centric programing"

أسلوب برمجة يحتم على الجمل أن تختار الكائن قبل تنفيذ أي إجراء عليه.

جلسة "session"

فاصل زمني يبدأ عند بداية تشغيل أكسس ثم الدخول وينتهي بالخروج، وفي حالة عدم وجود أي إجراءات تأمين يتم الدخول والخروج تلقائياً عند بداية تشغيل وإنهاء أكسس.

كائن مفرد "singular object"

كائن في نموذج كائنات ليس له ما يشبه ليتم تجميعه معه وذلك مثل كائن تطبيق.

صورة ثابتة "snapshot"

صورة ليست متغيرة لمجموعة من البيانات.

مجموعة سجلات من نوع صور ثابتة "snapshot-type recordset"

نسخة ثابتة لمجموعة من السجلات قائمة على جدول أو استعلام أو جملة SQL SELECT. في كائنات مجموعة السجلات من نوع صور ثابتة يحتوي الكائن على نسخة من السجل بأكمله "لا شيء سوى المراجع لحقول مذكرة وكائن OLE".

البرنامج المصدر "source code"

الجمل النصية التي تقوم بإدخالها في وحدة نمطية معينة.

قياس "standards"

مجموعة الخصائص المتعلقة بتقنية برمجية معينة مثل ActiveX أو ODBC.

جملة "statement"

وحدة كاملة من حيث البناء تعبر عن نوع معين من التشغيل والإعلان والتعريف. ويمكن تضمين أكثر من جملة في سطر واحد بشرط الفصل بينها بـ (:).

سلسلة "string"

نوع أساسي من البيانات يشتمل على معلومات في شكل حروف. ويمكن أن يحتوي متغير السلسلة على ٦٥٥٣٥ بايت، في كل بايت حرف واحد، وقد يكون ثابت أو متغير في الطول.

تعبير سلسلة "string expression"

أي تعبير يتم تقييمه على أنه سلسلة من الحروف المتجاورة مثل "& ", LastName & .FirstName

لغة استعلامات مبنية (SQL) "Structured Query Language"

اللغة الأساسية للتفاعل مع قاعدة بيانات علائقية، باستخدام SQL يمكن كتابة جمل لاسترداد ومعالجة مجموعة من السجلات في قاعدة البيانات.

إجراء فرعي "sub procedure"

إجراء يقوم بتنفيذ عملية و لا يرجع قيمة.

التزامن "synchronization"

عملية تحديث كائنان مرتبطان ارتباطا داخليا بحيث يعرضا معلومات دقيقة وحالية.

خطأ في بنية الجملة "syntax error"

خطأ في البنية التركيبية للتعليمات البرمجية، ويشمل أخطاء إملائية في الكلمات الأساسية والترقيم والمسافات وعدم تطابق الأقواس.

T.

مجموعة سجلات من نوع جداول "table-type recordset"

مجموعة من السجلات تمثل جدول واحد في ملف mdb.خاص بقاعدة البيانات المفتوحة وليس في جدول مرفق أو ODBC. وعند استخدام كائن مجموعة سجلات من نوع جداول فأنت تستخدم الجدول مباشرة. ويتوافر هذا الكائن مع مساحات عمل مايكروسوفت جيت فقط.

عملية "transaction"

مجموعة من التغييرات يتم إجراءها كوحدة واحدة، وعند معالجة العملية إما أن يتم إجراء كل التغييرات أو أن لا يتم إجراء أي تغيير.

خطأ قابل للتصيد "trappable error"

نوع معين من الأخطاء يمكن التعرف عليه وإعطاءه رقم يسمى "رمز الخطأ" ورسالة خطأ افتراضية من قبل أكسس أو VBA أو محرك قاعدة البيانات. وعند حدوث خطأ يقوم أحد إجراءات VBA بتحديد رمز الخطأ إذا كان موجوداً بالإضافة إلى تعليمات لتصحيحه.

بدء "trigger"

عمل ينتج عنه بداية إجراء. على سبيل المثال عند فتح نموذج يتم بدء حدث Open له.

مكتبة أنواع "type library"

ملف يشتمل على وسائل وخصائص وكائنات وثوابت خاصة بتطبيق آخر أو بقاعدة بيانات أكسس أخرى. ويمكن معالجة هذه الوسائل والخصائص والكائنات والثوابت باستخدام VBA.



عنصر تحكم غير منضم "unbound control"

عنصر تحكم في نموذج أو تقرير يكون إعداد خاصية ControlSource له ليس نفس اسم الحقل أو العمود. وعناصر التحكم غير المنضمة إما أن تتخذ من أحد التعبيرات كإعداد خاصية ControlSource لها أو إعداد خاصية ControlSource فارغة. ولا يوجد عنصر التحكم الغير منضم إلا أثناء فتح النموذج أو التقرير.

تحديث "update"

قبول وتخزين التغييرات التي تم إجرائها للبيانات في حقل أو سجل.

كائن معرف بواسطة المستخدم "user-defined object"

كائن مخصص ومعرف في الوحدة النمطية للفئات. ويمكن إنشاء مجموعة من الإجراءات الخاصة بوسائل وخصائص كائن جديد، كما يمكن إنشاء مثال للكائن ومعالجة المثال بهذه الوسائل والخصائص.



التحقق من الصحة "validation"

عملية مراجعة البيانات لتحديد ما إذا كانت القيمة تفي بالتحديدات والشروط المحددة.

متغير "variable"

موقع تخزين مؤقت في الذاكرة يمكن تسميته واستخدامه في حفظ قيمة أو في الإشارة إلى كائن معين.

متغير افتراضي "variant"

نوع بيانات افتراضي خاص بمتغيرات VBA عند عدم استخدام حرف إعلان النوع أو تحديد نوع البيانات أو نوع الكائن في جملة الإعلانات.



تعبير مراقبة "Watch expression"

تعبير يتم تحديده ليخضع للمراقبة في إطار المراقبات.

ملف معلومات مجموعة العمل "workgroup information file"

ملف قاعدة بيانات يقوم بتخزين معلومات عن المستخدمين في مجموعة العمل.

Z

سلسلة ذات طول صفري "zero-length string"

سلسلة لا تشتمل على أية حروف. ويمكن إدخال سلسلة ذات طول صفري بكتابة علامتي ترقيم بينهما مسافة.


Access 2000 VBA
Handbook
SYBEX
SYBEX, Inc.
may be copyrighted by
and have been so noted

Get Up to Speed on Access 2000 Functionality: ActiveX Data Objects, "Projects," and Other New Objects

Take Advantage of the Improved Visual Basic Editor and new Properties and Methods

Write VBA Procedures to Automate Any Task

Set Up Access to Work with Oracle, SQL Server, and Other Back-End Applications

Access 2000 VBA Handbook

Want to get the most out of Access? Then you need to know VBA.

Access 2000 is here! Never has the power of Access depended so much on VBA programming. This makes the **Access 2000 VBA Handbook** essential reading if you want to use Microsoft's popular database product to its full advantage. If you're an Access user, you'll benefit from step-by-step coverage of VBA basics that will simplify your work. If you're a developer, you'll master the high-end techniques introduced in the latest release—so you can stay on top of your job. You'll learn how to:

- Create databases and projects
- View and print reports
- Run queries to find specific records
- Automate complex and time-consuming procedures
- Synchronize forms and tables
- Understand the ADO and DAO models
- Customize the user interface
- Access data using OLE DB and ODBC
- Create VBA procedures for navigating a database or project
- Use object properties and methods to write VBA procedures
- Create relationships between forms
- Execute commands using SQL statements in VBA code
- Use ActiveX controls in your VBA procedures

Featured on the CD: On the enclosed CD, you'll find sample databases that illustrate many of the techniques covered inside. Also included are additional VBA reference materials, a Windows API file documenting Windows functions you can call using VBA, and the Event Logger database application for hands-on experience with events.

About the Author

Susann Novalis, Ph.D., is a Professor of Mathematics and Associate Dean of the College of Science and Engineering at San Francisco State University. She is the author of Automating Access with Macros, Mastering FrontPage 97, and Access 97 Macro & VBA Handbook, all from Sybex. She has also published articles in the magazines Access Visual Basic Advisor and Internet Advisor.



USER LEVEL	INTERMEDIATE/ADVANCED
BOOK TYPE	HOW-TO/REFERENCE
CATEGORY	APPLICATION DEVELOPMENT

ISBN 0-7821-2324-4



